全国第三届研究生数学建模竞赛

题 目　　学生面试中教师安排的优化与算法

摘　　要：

本文探讨高校自主招生的专家面试阶段，对面试教师安排的优化问题，主要考虑以下三个问题：

针对问题一，采用组合数学中的填充设计方法，对没有两位以及三位面试老师相同的约束条件分别给出 N、M 应当满足的不等式：

$$N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3} \right\rfloor \right\rfloor \text{和} N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\left\lfloor \frac{M-2}{2} \right\rfloor \right\rfloor \right\rfloor$$

针对问题二，本文结合问题一的结论以及对 $Y_1 \sim Y_4$ 的分析，认为 $Y_1 \sim Y_4$ 的要求可以通过考虑 Y_1, Y_2 及 Y_3'：每两位老师共同面试的学生数量应尽量均衡而基本满足。通过建立邻接矩阵 $A_{M\times N}$，并令 $B = AA^T, C = A^T A$，可以由矩阵 B、C 的元素实现 Y_1, Y_2 及 Y_3' 的量化。把 Y1、Y2 和 Y3’量化后作为优化目标，采用多目标规划遗传算法搜索最优解。通过染色体的竞争择优，从而保留较高适应度的个体，使整个进化过程朝着更优解的方向进行，最终可以根据需要达到不同程度优化的分配方案。对 N=379，M=24 的情形应用该算法给出了具体的面试老师分配方案，以及对该方案进行了评价。

针对问题三，面试组由两文两理老师组成，在此假设下，此时问题一给出的 N、M 应当满足的不等式修改为：$N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M}{4} \right\rfloor \right\rfloor$ 和 $N \le \frac{M-2}{2}\left\lfloor \frac{M}{4}\left\lfloor \frac{M}{4} \right\rfloor \right\rfloor$。对问题二结合问题一的结论知 N=379，M=24（文科、理科老师各 12 名）的情形，能够做到两个不同的“面试组”没有 3 个共同老师，并且实现了这一做法。

本文最后讨论以上模型的优缺点，探讨考生与面试老师之间分配的均匀性和面试公平性的关系并提出一些合理化建议。

关键词：区组设计，填充设计，多目标规划，遗传算法

参赛队号 10394003

参赛密码 _____

（由组委会填写）

一、问题的提出

自 2003 年以来，教育部为深化高校录取制度改革，进一步扩大了高校在招生中的自主权，北京大学、清华大学等 22 所高校实行了部分招生计划自主招生。2004 年，实行自主招生的高校数目进一步扩大到 28 所，选拔方式也进一步多样化。06 年 4 月复旦大学通过"面试"录取新生，彻底摒弃了已实施多年的高考笔试制度，由面试直接录取新生，这是对高校进一步扩大自主权改革的又一次全新探索与尝试。

当然，高校自主招生体系还处于探索阶段，还不完善，对于面试的考生，在全面衡量该考生的高中学习成绩及综合表现后通过采用专家面试的方式，决定录取与否。某高校在今年自主招生中，经过初选合格进入面试的考生有 N 人，拟聘请老师 M 人。每位学生要分别接受 4 位老师（简称该学生的“面试组”）的单独面试。面试时，各位老师独立地对考生提问并根据其回答问题的情况给出评分。由于这是一项主观性很强的评价工作，老师的专业、提问内容、提问方式以及评分习惯都会有较大差异，因此面试同一位考生的“面试组”的具体组成不同会对录取结果产生一定影响。为了确保面试工作的公平性，提出以下要求：

Y1. 每位老师面试的学生数量应尽量均衡；

Y2. 面试不同考生的“面试组”成员不能完全相同；

Y3. 两个考生的“面试组”中有两位或三位老师相同的情形尽量的少；

Y4. 被任意两位老师面试的两个学生集合中出现相同学生的人数尽量的少。

在满足 Y1～Y4 的某些条件下，考虑：

问题一：设考生数 N 已知，在满足 Y2 条件下，说明聘请老师数 M 至少分别应为多大，才能做到任两位学生的“面试组”都没有两位以及三位面试老师相同的情形。

问题二：根据 Y1～Y4 的要求建立学生与面试老师之间合理的分配模型，并就 N＝379，M＝24 的情形给出具体的分配方案（每位老师面试哪些学生）及该方案满足 Y1～Y4 这些要求的情况。

问题三：如果面试老师中理科与文科的老师各占一半，并且要求每位学生接受两位文科与两位理科老师的面试，并在此假设下分别回答问题一与问题二。

问题四：讨论考生与面试老师之间分配的均匀性和面试公平性的关系。为了保证面试的公平性，除了题目中提出的要求外，还有哪些重要因素需要考虑，试给出新的分配方案或建议。

二、模型分析与评价

问题一：聘请老师数 M 的下界

（一）问题一重述

考生数 N 已知，在满足 Y2（即面试不同考生的“面试组”成员不能完全相同）条件下，分别考虑任意两位学生的“面试组”必须保证没有两位或者三位面试老师相同的情形出现，聘请的老师数 M 至少应该多大。

（二）符号说明

v:聘请老师的个数

N：面试学生的个数

X：聘请的老师构成的集合 $X=\{1,\cdots,v\}$

B_i: 由 X 中的 k 个老师组成的子集，称之为区组，其中 b 为区组数目，

$$B_i=\{x_{i1},x_{i2},\cdots,x_{ik}\}\ ,x_{ij}\in\{1,\cdots,v\}\ (i=1,\cdots,b;j=1,\cdots,k)$$

$B_j^{'}$：区组 $B_i=(\underbrace{x_i,\overbrace{x_j\circ,\cdots,\circ}^{k-2个}}_{k个})$ 中形如 $\overbrace{(x_j,\circ,\cdots,\circ)}^{k-1}$ 的子区组

N_i：由 x_i 老师面试的学生个数

$N_{i,j}$：由 x_i,x_j 老师面试的学生个数

$\lambda(x_i,x_j)$：X 中任意一对老师（x_i,x_j）在区组中恰好同时出现的次数

$\{x_i,x_j\}$：i,j 两个老师组成的二元集 $(x_i,x_j\in\{1,\cdots,v\})$

$\{x_i,x_j,x_k\}$：i,j,k 两个老师组成的三元集 $(x_i,x_j,x_k\in\{1,\cdots,v\})$

$\lfloor x\rfloor$：表示对实数 x 下取整

（三）模型分析与求解

分别考虑任意两位学生的“面试组”保证没有两位或者三位相同面试老师的情形，这等价于任意两位学生的“面试组”至多只有相同的一位老师，或者任意两位学生的“面试组”至多只有相同的一位或者两位老师，我们的目标是在满足以上所需条件下使得聘请的老师数尽可能的少，我们采用的策略是均衡不完全区组设计（简称 BIBD）。

称有限集 X 上的一些 k 元子集 $B_i(1\le i\le b)$ 构成的族 B 为 X 上的一个区组设计，记为 $D=(X,B)$，X 称为此设计的基集，而子集族 B 中的诸子集 $B_i(1\le i\le b)$ 则称为此设计的区组。基集 X 中的元素个数 $|X|=v$ 称为设计的阶。称 k 为区组容量（或区组大小或区组长度）。对 $x\in X,B$ 中含有 x 的区组个数称为元素 x 的重复数，记做 $r(x)$。对 $x,y\in X$，且 $x\ne y$，B 中包含二元子集 $\{x,y\}$ 的区间个数称为元素的相遇数，记作 $\lambda(x,y)$。

我们知道一个(v,k,λ)—BIBD 要求任意一对点(x,y)恰好出现在λ个区组之中，这样的设计只在点数满足一定条件时才可能存在，也即：

$$\lambda(v-1)=0(\mathrm{mod}\,k-1)$$
$$\lambda v(v-1)=0(\mathrm{mod}\,k(k-1))$$

对于某些问题中需要考虑的点数v为任意取值的情形，则可以适当放宽点对“恰好”出现在λ个区组的条件，也即要求“至多”出现在λ个区组中（称为填充设计 **Packing Design**）。

⑴、对于任意两位学生的“面试组”保证没有两位相同面试老师的情形，必须要求任意一对老师$\{x_i,x_j\}$在区组$B_i(1\le i\le b)$中至多出现一次，且为了保证聘请的老师数M尽可能的少，我们遵循的一个原则是让任意一对老师$\{x_i,x_j\}$在区组$B_i(1\le i\le b)$中尽可能都出现一次，按 BIBD 策略，即λ=1，k=4，v=M 的填充设计。

我们先考虑更一般的情况，随意给定一面试老师$x_i\in X$，X是 M 元素集，区组设计(X,B)，设x_i元素在区组中出现的次数为$r=r(x_i)$，不妨假设这些区组为$B_1,B_2,\cdots,B_r$，形如：

$$B_1=(\underbrace{x_i,\overbrace{\circ,\cdots,\circ}^{k-1个}}_{k个})$$
$$\cdots\cdots$$
$$B_r=(\underbrace{x_i,\overbrace{\circ,\cdots,\circ}^{k-1个}}_{k个})$$

其中“∘”代表除了x_i的其他$v-1$个面试老师，且两两互不相同。

由于任意一对老师$\{x_i,x_j\}(i\ne j)$在区组$B_i(1\le i\le b)$中至多出现λ次，用这$v-1$个面试老师填充$B_i(i=1,\cdots,r)$中的圆圈“∘”，则X中x_i以外的$v-1$个面试老师出现在这r组中的次数为$\lambda(v-1)$，则包含x_i元素的区组中至多出现的次数，即由x_i老师面试的学生个数N_i至多为：

$$\left\lfloor\frac{\lambda(v-1)}{k-1}\right\rfloor$$

面试老师个数为v，即x_i有v种取法，考虑到每个区组重复计算k次，因此

最大可能的填充数为：

$$\left\lfloor \frac{v}{k}\left\lfloor \frac{\lambda(v-1)}{k-1}\right\rfloor \right\rfloor 。$$

特别地，问题一考虑λ=1，k=4，v=M 的情况，即满足我们题目中所要求的约束条件，则可得到任两位面试老师不重复的 4 元组的个数至多为：

$$\left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\right\rfloor \right\rfloor$$

又因为每一个学生的面试有四个老师的参与，也即一个四元老师组对应着一个学生，则：

$$N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\right\rfloor \right\rfloor$$

综上分析，在满足$Y2$条件下，确保任两位学生的面试组都没有两位面试老师相同，面试老师数M必须满足不等式：

$$N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\right\rfloor \right\rfloor \qquad \cdots\cdots \quad (1)$$

当学生数N给定时，可得到M的下界。

⑵、对于任意两位学生的“面试组”保证没有三位相同面试老师的情形，必须要求任意 3 位老师$\{x_i, x_j, x_k\}$在区组$B_i(1 \le i \le b)$中至多出现一次，且为了保证聘请的老师数M尽可能的少，我们遵循的一个原则是让任意 3 位老师$\{x_i, x_j, x_k\}$在区组$B_i(1 \le i \le b)$中尽可能都出现一次，按 BIBD 策略，即参数λ, k, v满足λ=1，k=4，v=M 的填充设计。

同样我们先考虑更一般的情况，X是包含 M 个老师的 M 元集，区组设计(X, B)，随意先给定一面试老师$x_i \in X$，有$v-1$种取法，然后再确定另一面试老师$x_j \in X$，有$v-2$种取法，设x_i, x_j两元素在区组中出现的次数为$r = r(x_i, x_j)$，不妨假设为$B_1, B_2, \cdots, B_r$：

$$B_1 = (\underbrace{x_i, x_j \overbrace{\circ, \cdots, \circ}^{k-2\text{个}}}_{k\text{个}});$$

……

$$B_r=(\underbrace{x_i,x_j,\overbrace{\circ,\cdots,\circ}^{k-2个}}_{k个});$$

其中“$\circ$”代表除了x_i,x_j面试老师的其他$v-2$个面试老师，且两两互不相同。

由于任意一对老师$\{x_i,x_j,x_k\}(i,j,k互不相同)$在区组$B_i(1\le i\le b)$中至多出现$\lambda$次，用剩下的$v-2$个面试老师填充$B_i(i=1,\cdots,r)$中的圆圈“$\circ$”，则$X$中$x_i,x_j$以外的$v-2$个老师出现在这$r$组中的次数为$\lambda(v-2)$，则包含$x_i,x_j$元素的区组中至多出现的次数，即由$x_i,x_j$老师面试的学生个数$N_{i,j}$至多为：

$$\left\lfloor\frac{\lambda(v-2)}{k-2}\right\rfloor$$

对于面试老师$x_j\in X$，在确定$x_i\in X$后，面试老师x_j则有$v-1$种取法：

$$B_1=(\underbrace{x_i,\overbrace{x_j,\circ,\cdots,\circ}^{k-1个}}_{k个})$$

……

$$B_r=(\underbrace{x_i,\overbrace{x_j,\circ,\cdots,\circ}^{k-1个}}_{k个})$$

不妨设形如$(\overbrace{x_j,\circ,\cdots,\circ}^{k-1})$的$k-1$元集记做区组$B_i(1\le i\le r)$的子区组$B_j'(1\le j\le r)$

$$B_1'=(\overbrace{x_j,\circ,\cdots,\circ}^{k-1})$$

……

$$B_r'=(\overbrace{x_j,\circ,\cdots,\circ}^{k-1})$$

其中，$B_j'(1\le j\le r)$的元素是取自区组$B_i(1\le i\le r)$的后$k-1$个元素。

因此对于区组$B_i(1\le i\le r)$的子区组$B_j'(1\le j\le r)$最大可能的填充数为：

$$\left\lfloor\frac{v-1}{k-1}\left\lfloor\frac{\lambda(v-2)}{k-2}\right\rfloor\right\rfloor$$

而x_i有v种取法，因此最大可能的填充数为：

$$\left\lfloor\frac{v}{k}\left\lfloor\frac{v-1}{k-1}\left\lfloor\frac{\lambda(v-2)}{k-2}\right\rfloor\right\rfloor\right\rfloor。$$

特别地，问题一考虑λ=1，k=4，v= M 的情况，即满足我们题目中所要求的约束条件，则可得到任三位面试老师不重复的 4 元组的个数至多为：

$$\left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\left\lfloor \frac{M-2}{2}\right\rfloor\right\rfloor\right\rfloor$$

同样地，每一个学生的面试有四个老师的参与，也即一个 4 元组对应着一个学生，即：

$$N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\left\lfloor \frac{M-2}{2}\right\rfloor\right\rfloor\right\rfloor 。$$

综合分析，在满足$Y2$条件下，确保任两位学生的面试组都没有三位面试老师相同，面试老师数M必须满足不等式：

$$N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\left\lfloor \frac{M-2}{2}\right\rfloor\right\rfloor\right\rfloor \qquad \cdots\cdots(2)$$

当学生数N给定时，可得到M的下界。

问题二：约束条件下学生与面试老师之间分配模型

（一）问题二重述

问题二要求我们满足四个约束条件(Y1～Y4)下,,建立学生与面试老师之间的分配模型，并就 N=379，M=24 的情形给出具体的分配方案（每位老师面试哪些学生）及该方案满足 Y1～Y4 这些要求的情况。

Y1. 每位老师面试的学生数量应尽量均衡；

Y2. 面试不同考生的“面试组”成员不能完全相同；

Y3. 两个考生的“面试组”中有两位或三位老师相同的情形尽量的少；

Y4. 被任意两位老师面试的两个学生集合中出现相同学生的人数尽量的少。

（二）模型分析

根据问题二的阐述，可以清晰地知道学生与面试老师之间必然要建立一对四的面试关系，也即在学生与面试老师之间有着多种分配关系，在此，我们寻求的是一种更加合理的分配关系，使得这种分配关系满足或者说越逼近于我们目标：

以$M=24, N=379$为例，若要任两位学生的“面试组”中都没有两位老师相同，则由问题一的不等式（1）可得学生数至多为$\left\lfloor \frac{24}{4}\left\lfloor \frac{23}{3}\right\rfloor\right\rfloor = 42$，而实际上$N=379$。所以肯定出现两个考生的“面试组”中有两位老师相同的情形，我们的优化目标是所有两位老师重复出现的次数总量尽可能的少。

如果要任意两位考生的面试老师组中都没有三位老师相同，则由问题一的不等式（2）可知学生数至多可为 $\left\lfloor \frac{24}{4}\left\lfloor \frac{23}{3}\left\lfloor \frac{22}{2} \right\rfloor \right\rfloor \right\rfloor = 504$，而实际考生数 $N = 379$，因此理论上可以做到面试老师组中都没有三位老师相同的情形。但通常在实际安排中难以做到，因此让三位老师相同的情形尽量少依然是个优化目标。

进一步分析可知道，实现两位老师重复出现的次数总是尽可能少的通常采用的策略是任两位老师重复出现的次数尽量均衡。而三位老师每重复出现一次，则对两位老师重复出现的次数总是增加 3，因此两位老师重复出现的次数，总量尽可能少的情形通常也是发生在三位老师重复出现的次数总量较小的情形。

对于条件 Y4，老师 i, j 面试的学生集合中出现相同的学生，就是老师 i, j 出现的同一面试组，因此老师 i, j 面试学生集合中出现相同的学生人数就是老师 i, j 重复出现在“面试组”中的重复数。要做到任意两位老师面世相同学生人数尽量少，仍然是使得任意两位老师重复出现的次数尽量均衡。

综上所述，我们的优化策略为以下三个目标：Y_1, Y_2, Y_3'

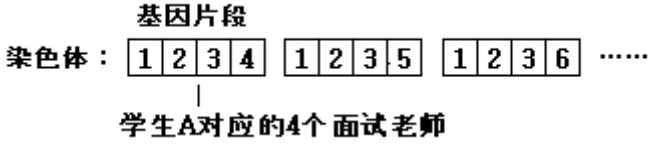
其中 Y_3'：任意两位老师重复出现在一个面试组中的重复数尽可能均衡。

（三）模型求解

为了使得我们的分配方案更加合理，找到满足这些目标的最佳设计方案，我们采取的策略是：目标规划遗传算法。借助遗传算法实施来解决多目标优化问题（Multi-objective Optimization Problem），使整个过程朝着更优解的方向进行。

1、遗传编码设计

本文将面试不同考生的“面试组”4 个成员做为一个基因片段，基因片段的异同与基因片段中数字的排列无关（例如：两个基因片段 1 2 3 4 与 1 3 2 4 在本文认为是一样的）。基因片段中的单个基因为面试老师的序号，从而按照学生序列构造由面试老师序号组成的染色体。

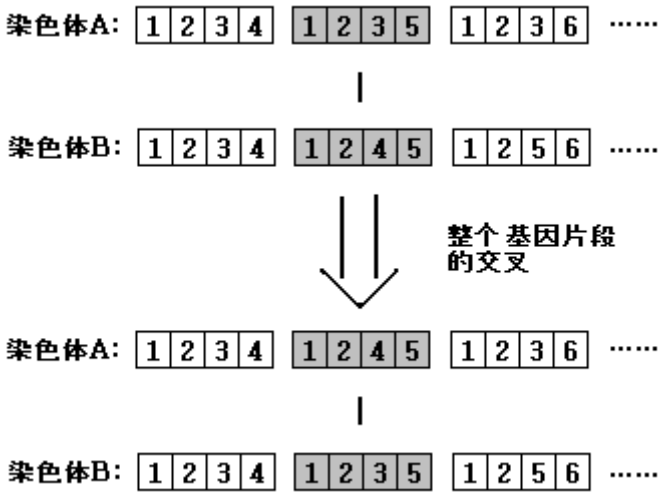


2、初始种群的产生

由于题目 Y2 条件对每个学生遇到的面试老师做了限制（面试不同考生的“面试组”成员不能完全相同），染色体的构成也因此有了最低的限制条件，因此不能采用随机产生染色体的方法来建立初始种群。所以本文先通过贪婪算法搜索出几个仅符合 Y2 条件的染色体，构造出初始种群。

3、交叉算子

传统遗传算法中的交叉运算是对单独的基因进行交叉，但由于在本题中染色的特征是由“面试组”中的 4 个成员构成的基因片段，所以本文在交叉运算中实行对基因片段的交叉而不是单独基因的交叉。



同时根据 Y2 的要求，染色体 A 中要交叉的基因片段在不能与染色体 B 中任何基因片段相同，染色体 B 中要交叉的基因片段在不能与染色体 A 中任何基因片段相同。

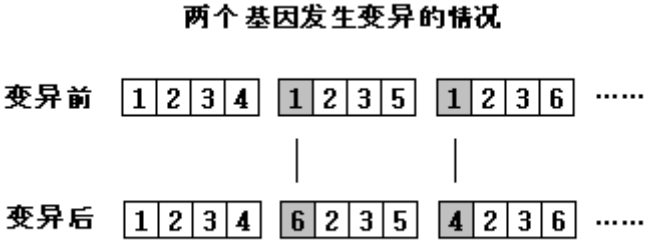
本文采用多点交叉的策略，本文中采用动态的交叉概率 P_c 控制交叉操作的使用频率。

4、变异算子

本文采用基本位变异（Simple Mutation）的方式来实现变异运算，以变异概率 P_m 指定一位或者某几位基因座上的值座变异运算，其具体操作过程如下：

（1） 对染色体上的每一个基因座，以变异概率指定它为变异基因。

（2） 对每一个变异基因，随机产生[1..M]中的数值，从而产生新的基因片段（该基因片段不能与该染色体中其它的基因片段相同，基因片段中的基因不能重复）。



5、适应度函数 F

通过一个染色体中的基因序列，即一分配方案，可建立如下邻接矩阵：

$$A=\left(a_{ij}\right)_{M\times N}$$

$$a_{ij}=\begin{cases}1, \text{第}i\text{个老师与第j个学生有面试关系}\\ 0, otherwise\end{cases}$$

$$1\le i\le M, 1\le j\le N$$

令 $B=AA^T=\left(b_{ij}\right)_{M\times M}$，$C=A^TA=\left(c_{ij}\right)_{N\times N}$，其中 $b_{ij}=\sum_{k=1}^{N}a_{ik}\cdot a_{jk}$，$c_{ij}=\sum_{k=1}^{M}a_{ki}\cdot a_{kj}$

矩阵 B,C 相应元素 b_{ij},c_{ij} 的涵义如下：

$b_{ii}\left(i=1\cdots M\right)$：第 i 号老师面试学生数；

$b_{ij}\left(i\ne j\right)$：第 i 号老师和第 j 号老师同组重复数；

$c_{ii}\left(i=1\cdots N\right)$：第 i 号学生的“面试组”老师个数，恒为 4；

$c_{ij}\left(i\ne j\right)$：第 i 号学生和第 j 号学生“面试组”相同老师的个数；

则 Y_1,Y_2,Y_3' 条件可以用矩阵 B,C 的元素来进行描述：

（1） Y1. 每位老师面试的学生数量应尽量均衡；

$$\Updownarrow$$

$b_{ii}\left(1\le i\le M\right)$ 的方差 $f_1\left(\alpha\right)$ 尽量小，即

$f_1\left(\alpha\right)=\frac{1}{M}\sum_{i=1}^{M}\left(b_{ii}-\frac{1}{M}\sum_{i=1}^{M}b_{ii}\right)^2=\frac{1}{M}\sum_{i=1}^{M}\left(b_{ii}-\frac{4N}{M}\right)^2$ 尽量小，其中 α 表示遗传的迭代次数；

（2） Y2 面试不同考生的“面试组”成员不能完全相同，即第 i 号学生和第 j 号学生“面试组”相同老师的个数小于等于 3

$$\Updownarrow$$

$$c_{ij}\left(i\ne j\right)<4;$$

（3） Y_3' ：任意两位老师重复出现在一个面试组中的重复数尽可能均衡

$$\Updownarrow$$

$$f_2(\alpha)=\frac{1}{C_M^2}\sum_{i<j}(b_{ij}-\frac{6N}{C_M^2})^2$$ 的值尽可能小。

则适应度函数F可描述为：

$$F=w_1\frac{f_1(\alpha)}{f_1(\alpha-1)}+w_2\frac{f_2(\alpha)}{f_2(\alpha-1)}$$

其中w_1、w_2为相应的权系数，可根据对Y_1, Y_3'三个要求条件的重视程度赋予w_1、w_2不同大小的数值，总和为 1 保持不变。

6、选择算子

本文采用竞争择优的选择方法，从而保留后代中较高适应度的个体，使整个进化过程朝着更优解的方向进行。

7、控制参数

（1）交叉概率P_c

在进化过程中，P_c始终控制着在遗传过程中起主导作用的交叉算子。较大的P_c 可以使各代充分相交，个体结构被破坏的可能性也越大，不能有效保护好的基因模式。较低的P_c 可以保持一个连续的解空间，但是进化的速度慢。

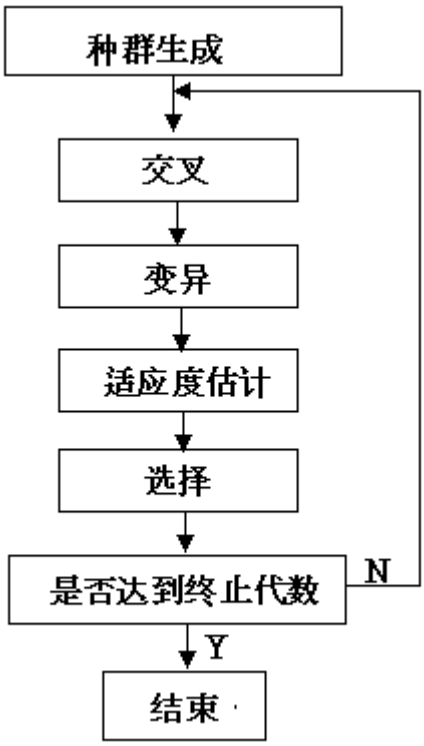
（2）变异概率P_m

本文中较大的P_m可以产生较多的新基因片段，增加了群体的多样性，但也有可能破坏掉很多好的模式。P_m太小则容易陷入局部极值，使解过早收敛产生“早熟”问题。

（3）控制方法

为了保证遗传算法的全局收敛性，就要维持解群体中个体的多样性 ,避免有效基因的丢失 。另一方面，为了加快收敛速度，就要使解群体较快地向最优状态转移，这又会减少群体多样性，容易陷入局部最优。因此，本文将进化过程划分为突和渐进两个阶段。由于在一开始采用贪婪法产生的种群有着很大的相似性，所以要将交叉概率P_c和变异概率P_m设置为较大的数值，本文中将P_c和P_m的初始数值设置为 0.6 与 0.3。通过较频繁的变异和交叉产生新的基因片段和染色体，扩大解的搜索空间避免陷入局部极值。之后随着进化的过程逐渐调小P_c与P_m的值，使其加快收敛速度。

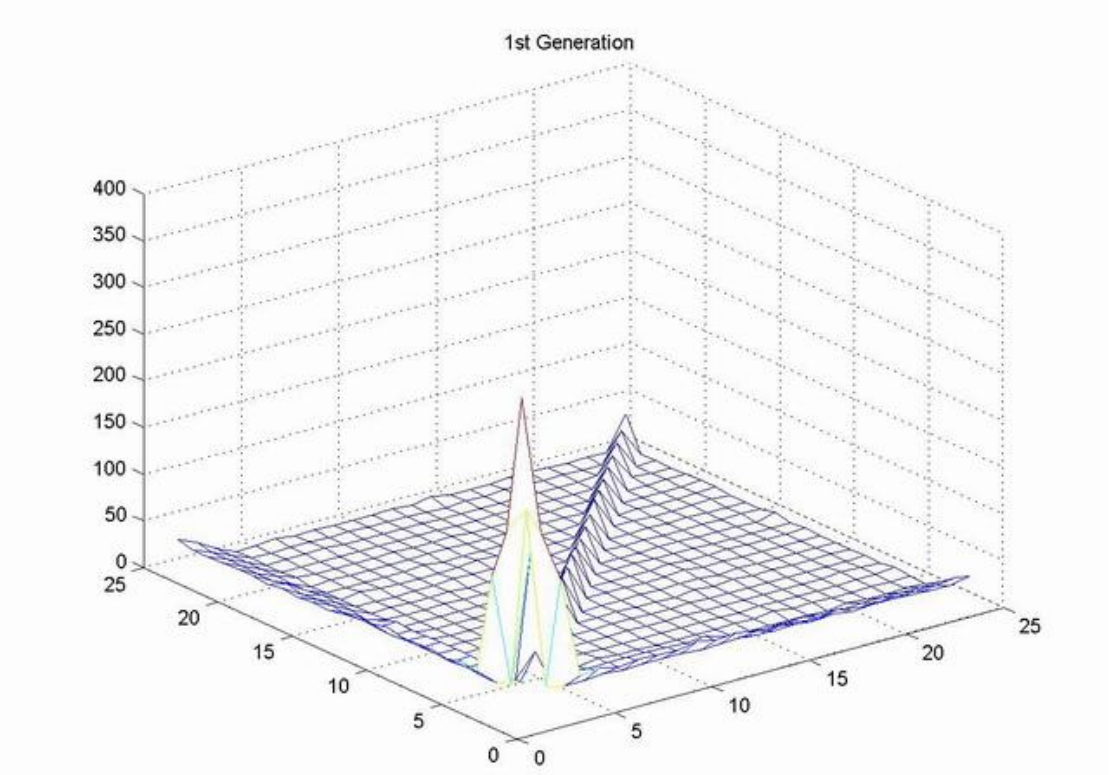
8、流程描述（具体程序见附录一）

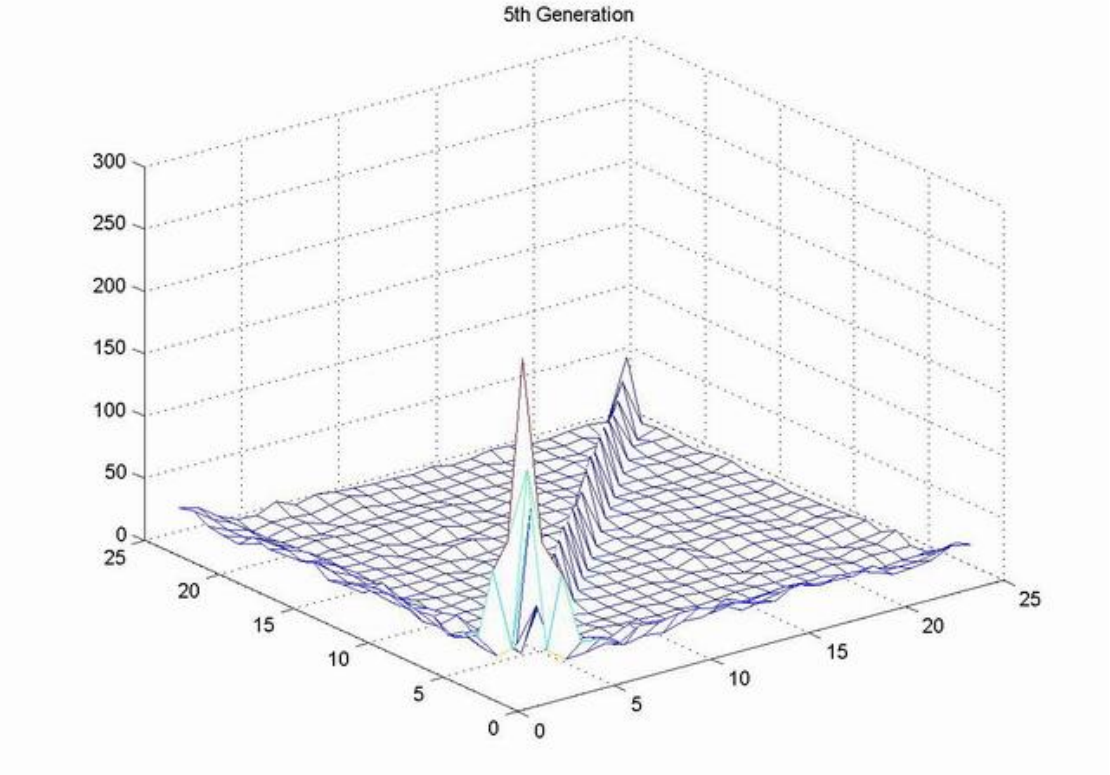


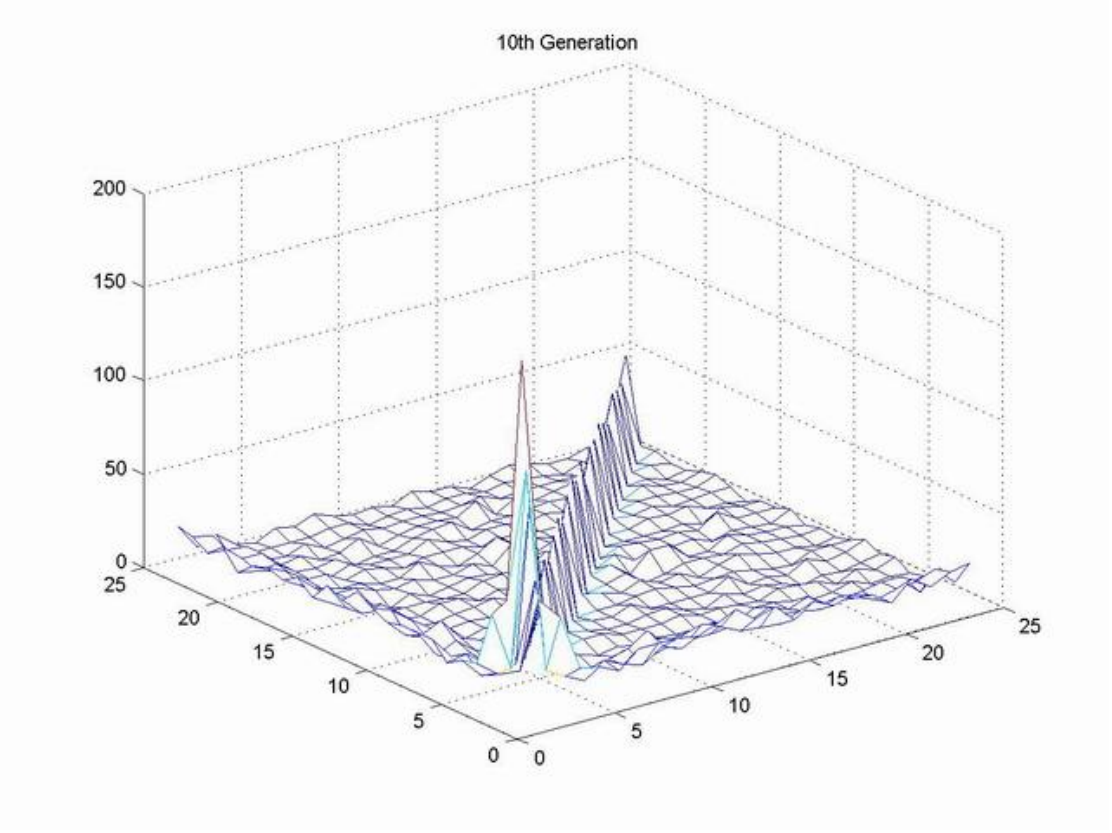
9、实验结果及评价

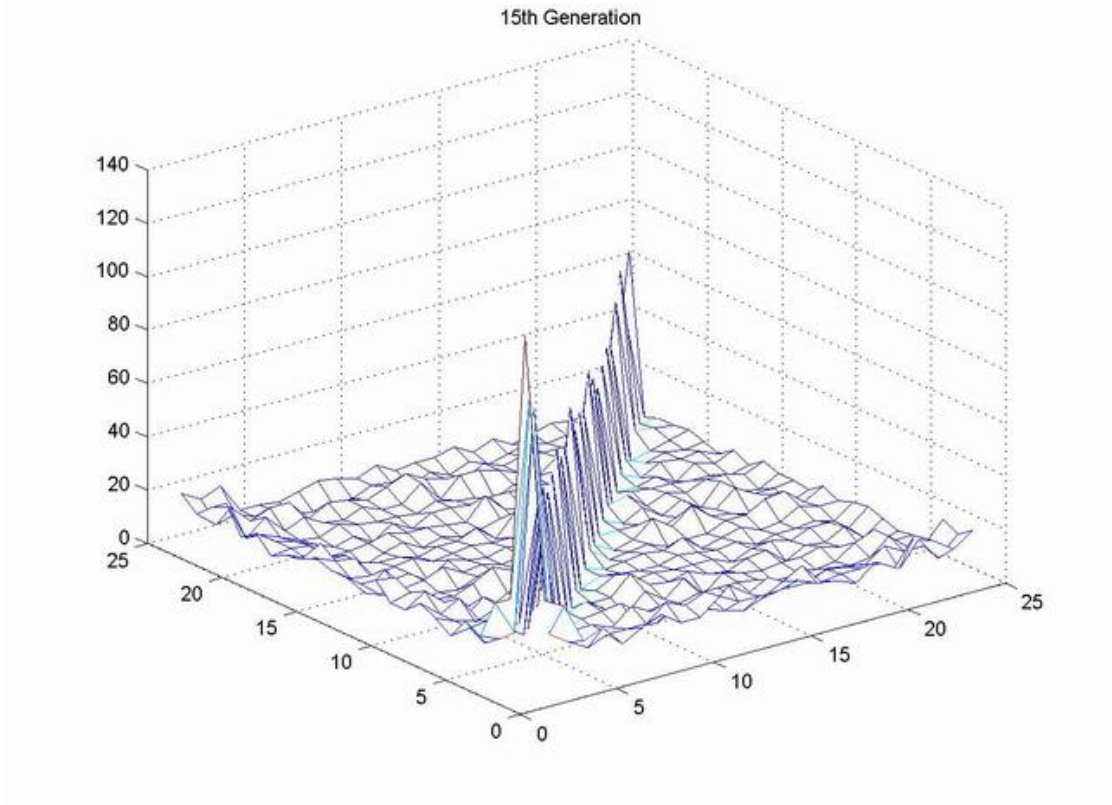
（1） 从 B 矩阵（ AA^T ）观察分析

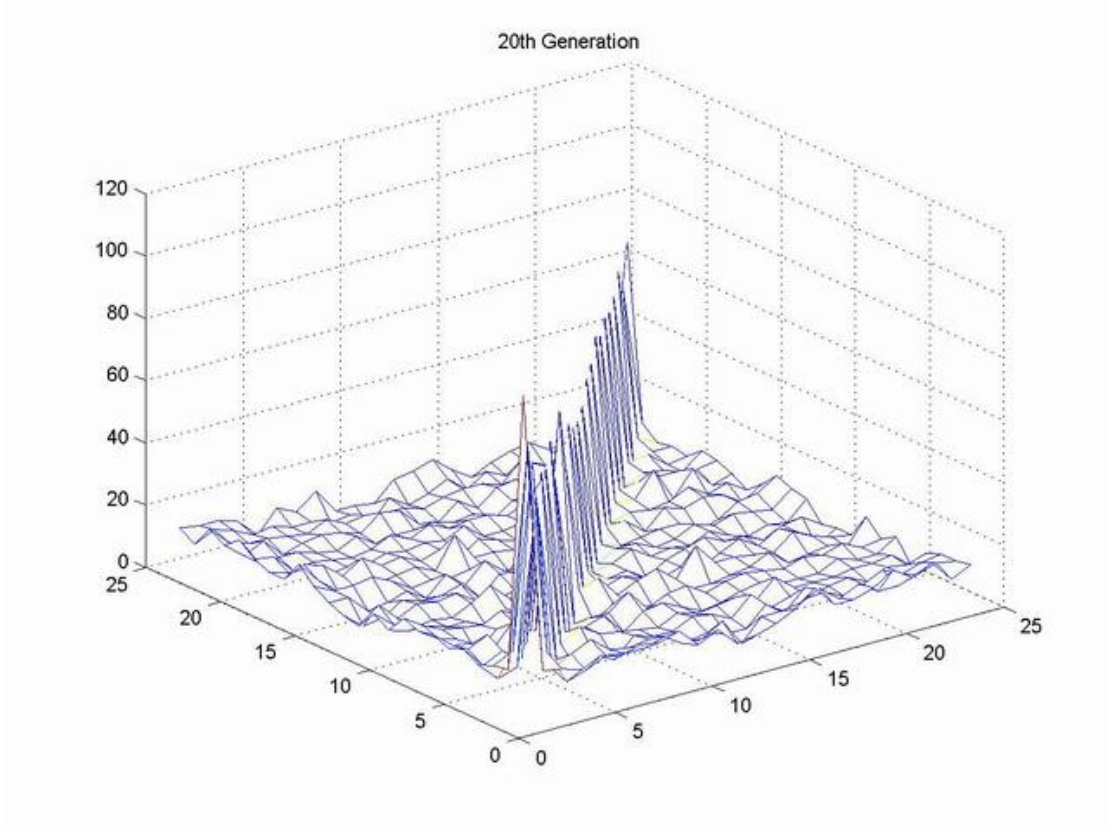
下面给出 1、5、10、15、20、25、30 代中较高适应度的染色体对应的 B 矩阵（见附录二）图形（水平坐标为老师序号，垂直坐标为学生数目）：

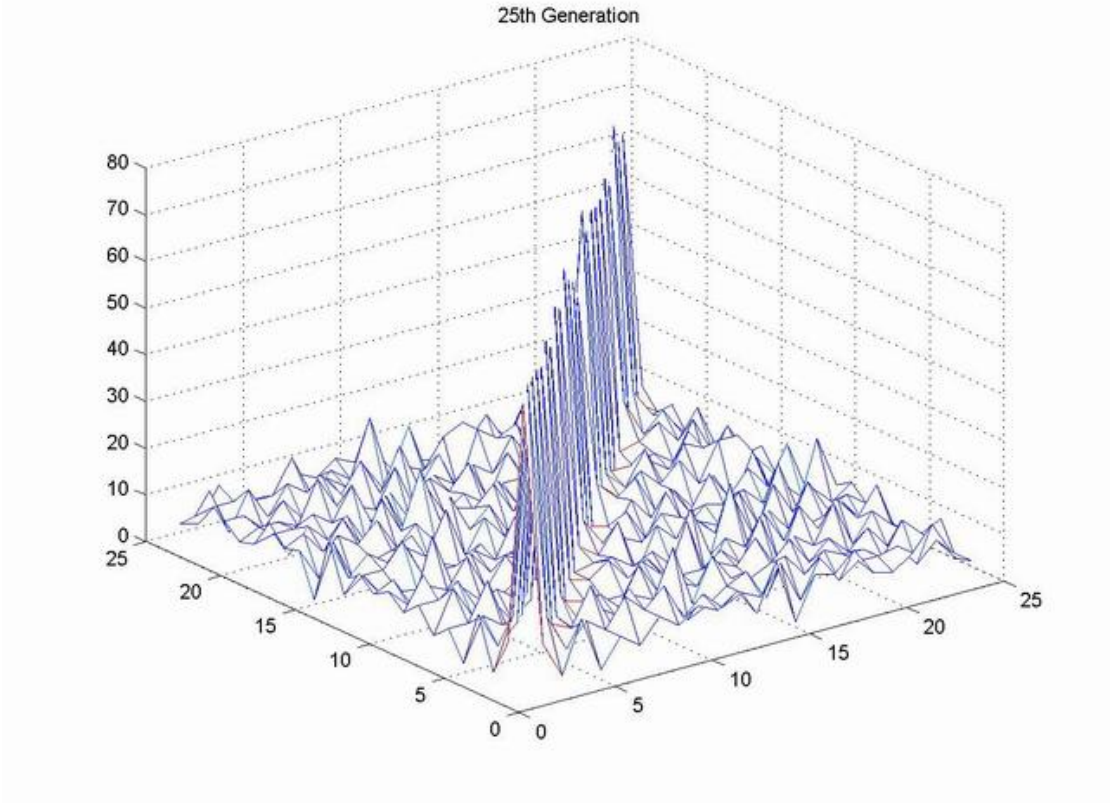

1st Generation
400
350
300
250
200
150
100
50
0
25
20
15
10
5
0
0
5
10
15
20
25

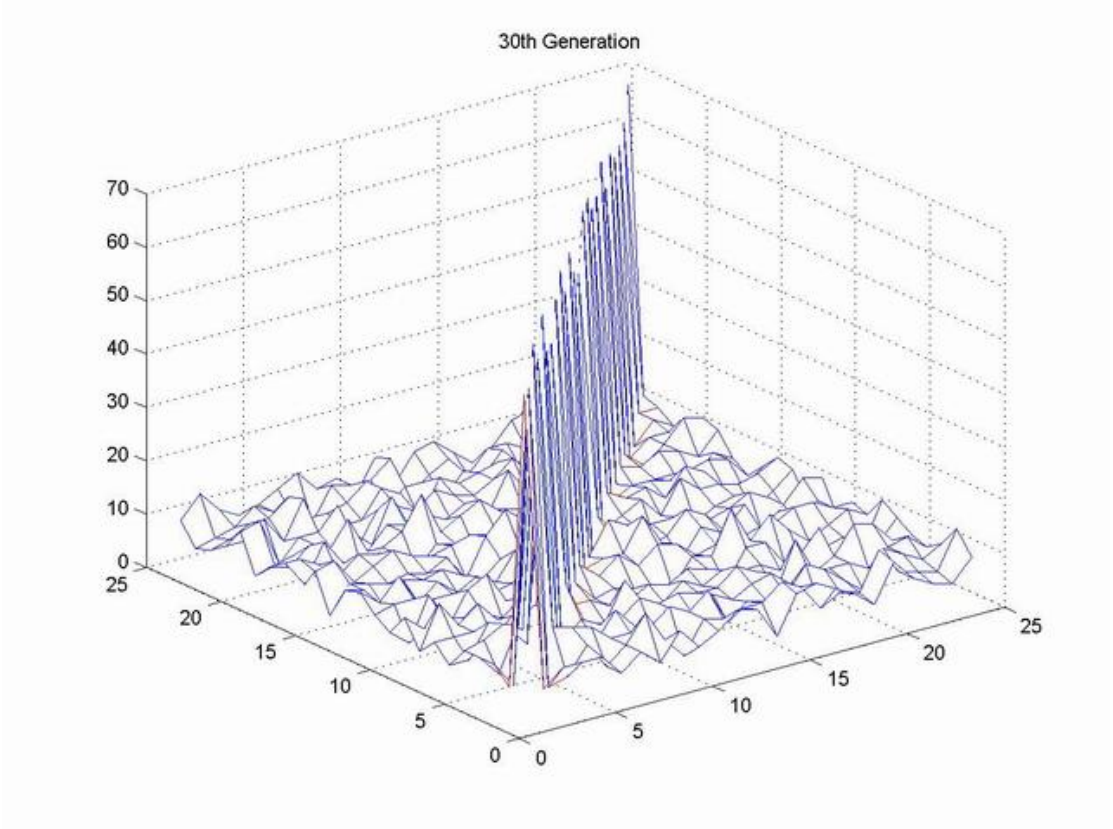

5th Generation
300
250
200
150
100
50
0
25
20
15
10
5
0
0
5
10
15
20
25


10th Generation
200
150
100
50
0
25
20
15
10
5
0
0
5
10
15
20
25


15th Generation
140
120
100
80
60
40
20
0
25
20
15
10
5
0
0
5
10
15
20
25


20th Generation
120
100
80
60
40
20
0
25
20
15
10
5
0
0
5
10
15
20
25


25th Generation
80
70
60
50
40
30
20
10
0
25
20
15
10
5
0
0
5
10
15
20
25



从以上进化过程看出，矩阵的对角元即各位老师面试的学生数目趋于均衡，矩阵的非对角元即不同老师面试的共同学生数目也趋于均衡。从而说明我们设置的等价限制条件是有效的，模型的建立是合理的。

（2） 从 C 矩阵观察，并结合问题一的结论分析说明：

将 M=24 代入下式得，学生数 N＝379 满足不等式：

$$42=\left\lfloor\frac{M}{4}\left\lfloor\frac{M-1}{3}\right\rfloor\right\rfloor\le N\le\left\lfloor\frac{M}{4}\left\lfloor\frac{M-1}{3}\left\lfloor\frac{M-2}{2}\right\rfloor\right\rfloor\right\rfloor=504$$

所以无法满足每两位学生的“面试组”中共同老师不超过 1，但可以满足共同老师数不超过 2。从下面给出的遗传算法中 1 至 30 代染色体对应的 C 矩阵（A^TA）中 0、1、2、3（从左至右）出现的次数及所占的比例中，可看出 3 出现的频率逐代降低并趋近于 0，也就是说是已经逼近最优解。

5678 [3.95%]	60928 [42.42%]	65572 [45.65%]	11084 [7.72%]
12738 [8.87%]	67674 [47.11%]	54598 [38.01%]	8252 [5.74%]
21176 [14.74%]	68538 [47.71%]	46924 [32.67%]	6624 [4.61%]
24778 [17.25%]	71458 [49.75%]	41650 [29.00%]	5376 [3.74%]
29630 [20.63%]	72238 [50.29%]	37012 [25.77%]	4382 [3.05%]
33534 [23.35%]	72196 [50.26%]	33778 [23.52%]	3754 [2.61%]
39496 [27.50%]	70496 [49.08%]	30048 [20.92%]	3222 [2.24%]
40446 [28.16%]	70300 [48.94%]	29416 [20.48%]	3100 [2.16%]
45450 [31.64%]	69412 [48.32%]	25848 [17.99%]	2552 [1.78%]

49800 [34.67%]	67656 [47.10%]	23580 [16.42%]	2226 [1.55%]
53350 [37.14%]	66462 [46.27%]	21494 [14.96%]	1956 [1.36%]
55516 [38.65%]	65722 [45.75%]	20260 [14.10%]	1764 [1.23%]
57646 [40.13%]	65098 [45.32%]	18918 [13.17%]	1600 [1.11%]
58640 [40.82%]	64582 [44.96%]	18496 [12.88%]	1544 [1.07%]
59638 [41.52%]	64224 [44.71%]	17974 [12.51%]	1426 [0.99%]
60474 [42.10%]	63708 [44.35%]	17730 [12.34%]	1350 [0.94%]
61186 [42.60%]	63460 [44.18%]	17264 [12.02%]	1352 [0.94%]
62220 [43.32%]	63040 [43.89%]	16766 [11.67%]	1236 [0.86%]
63110 [43.94%]	62440 [43.47%]	16448 [11.45%]	1264 [0.88%]
63252 [44.03%]	62520 [43.53%]	16264 [11.32%]	1226 [0.85%]
64336 [44.79%]	62090 [43.23%]	15694 [10.93%]	1142 [0.80%]
65050 [45.29%]	61478 [42.80%]	15688 [10.92%]	1046 [0.73%]
64954 [45.22%]	61888 [43.09%]	15348 [10.68%]	1072 [0.75%]
65348 [45.49%]	61606 [42.89%]	15248 [10.62%]	1060 [0.74%]
65532 [45.62%]	61516 [42.83%]	15092 [10.51%]	1122 [0.78%]
65604 [45.67%]	61534 [42.84%]	15060 [10.48%]	1064 [0.74%]
65808 [45.81%]	61278 [42.66%]	15124 [10.53%]	1052 [0.73%]
65668 [45.72%]	61444 [42.78%]	15116 [10.52%]	1034 [0.72%]
65864 [45.85%]	61226 [42.62%]	15116 [10.52%]	1056 [0.74%]
65874 [45.86%]	61248 [42.64%]	15170 [10.56%]	970 [0.68%]

具体分配方案（每位老师面试哪些学生）见附录三。

问题三：文理约束下聘请老师 M 的下界

（一）问题三重述

在满足问题一的基础上，增加一个约束，即在所有的面试老师中，理科与文科的老师各占一半，且每位学生都接受两位文科与两位理科老师的面试。在这种情形下，聘请的老师数 M （M 为偶数）应该满足什么样的条件。

（二）模型分析与求解

学生数 N 已知，在满足 $Y2$（即面试不同考生的“面试组”成员不能完全相同）及要求每位学生都要接受两文、两理老师面试条件下，考虑任意两位学生的“面试组”保证没有两位或者三位相同面试老师的情形，这等价于任意两位学生的“面试组”至多只有一位（一位或者两位）老师相同（有可能是文科老师也有可能是理科老师），我们的目标是在满足以上所需条件下使得聘请的老师数尽可能的少，我们采用的主体思想依旧是问题一的“均衡不完全区组设计”（简称 BIBD），并在此基础上进一步深化。

⑴、对于任意两位学生的“面试组”保证没有两位相同面试老师的情形，必须要求任意一对老师 $\{x_i, x_j\}$ 在区组 $B_i(1 \le i \le b)$ 中至多出现一次，先取定老师 x_i，共有 M 种取法，不妨设 x_i 为理科老师，并设 x_i 元素在区组中出现的次数为

$r=r(x_i)$，令这些区组为 $B_1,B_2,\cdots,B_r$：

$$B_1=(\underbrace{\overbrace{x_i}^{理},\circ,\circ,\circ}_{4})$$

……

$$B_r=(\underbrace{\overbrace{x_i}^{理},\circ,\circ,\circ}_{4})$$

（其中“$\circ$”代表所需的三个面试老师），则组成一个 4 元面试组还需三个面试老师，且这三位面试老师必须用除了 x_i 的两文一理来填充。由题中假设知，面试老师中文科理科老师各占一半，则一共可以分成 $\left\lfloor\frac{M}{4}\right\rfloor$ 组，在取定 x_i 后，理科老师还剩 $\frac{M}{2}-1$ 个，一共可以分成 $\frac{M}{2}-1$ 组，则包含理科老师 x_i 的区组至多出现的次数为：

$$\min\left\{\left\lfloor\frac{M}{4}\right\rfloor,\frac{M-2}{2}\right\}=\left\lfloor\frac{M}{4}\right\rfloor$$

即由 x_i 理科老师面试的学生个数 N_i 至多为：

$$\left\lfloor\frac{M}{4}\right\rfloor$$

由于 x_i 的取法数为 $\frac{M}{2}$，则可得到任两位面试老师不重复的 4 元组的个数至多为：

$$\left\lfloor\frac{M}{4}\left\lfloor\frac{M}{4}\right\rfloor\right\rfloor$$

因此，确保任意两位学生的面试组都没有两位面试老师相同，面试老师数 M 必须满足不等式：

$$N\le\left\lfloor\frac{M}{4}\left\lfloor\frac{M}{4}\right\rfloor\right\rfloor \qquad (3)$$

⑵、考虑对于任意两位学生的“面试组”，没有三位面试老师相同的情形。

符号说明：

不妨先任意取定理科老师 x,y,则要组成一个 4 元“面试组”，还需两位文科老师的参与，由于文科老师数为 $\frac{M}{2}$，将 $\frac{M}{2}$ 个文科老师中按“二元素对”进行划分，并且“二元素对”之间元素要求两两不同，则有 $\left\lfloor\frac{M}{4}\right\rfloor$ 对；接下来，将 y 老

师换成另一位理科老师z，即取定的老师变成x,z，同样的，可以找另一种文科老师的组合方式$\left\lfloor \frac{M}{4} \right\rfloor$组，……这个过程继续下去，直到取完所有的理科老师，则包含面试老师x且至多只有两位老师相同的“面试组”数为：

$$\left\lfloor \frac{M}{4} \right\rfloor \cdot \left(\frac{M}{2} - 1 \right)$$

又x可以是理科老师（总数为$\frac{M}{2}$）中的任何一个，过滤掉重复的情况，则满足至多两位老师相同的“面试组”个数最多为：

$$\left\lfloor \frac{M}{4} \cdot \left\lfloor \frac{M}{4} \right\rfloor \right\rfloor \left(\frac{M}{2} - 1 \right)$$

综合分析，在满足$Y2$条件下，确保任两位学生的面试组都没有三位面试老师相同，面试老师数M必须满足不等式:

$$N \le \left\lfloor \frac{M}{4} \cdot \left\lfloor \frac{M}{4} \right\rfloor \right\rfloor \left(\frac{M}{2} - 1 \right) \qquad (4)$$

下面给出面试老师数$M=12$的计算例子：

面试老师总数$M=12$，文科理科老师各占一半，1，2，…，6 代表 6 位理科老师，a,b,…,f 代表 6 位文科老师，不妨设任意取定的理科老师x=1，则在含老师x（即 1）的“面试组”中，满足至多只有两位面试老师相同的“面试组”有：$C_{\frac{12}{2}}^{2}=15$组，用其它理科老师更换x（即 1），则可得到另外 30 组。事实上，在下述表格中，将文科老师对固定，如(a，b)。可将理科老师对(1,2)换成(3,4),(5,6),在这一过程中，“面试组”数量增加到原来的三倍，因此满足至多两位老师相同的“面试组”有$C_{\frac{12}{2}}^{2}\times 3=45$组（具体安排列表如下）：

12	ab								
	cd								
	ef								
13	ac	23	af						
	be		bc						
	df		de						
14	ad	24	ae	34	ab				
	bf		bd		cd				
	ce		cf		ef				
15	ae	25	ac	35	ad	45	af		
	bd		be		bf		bc		
	cf		df		ce		de		
16	af	26	ad	36	ae	46	ac	56	ab
	bc		bf		ed		be		cd

	de		ce		cf		df		ef

表一（其中“12ab”代表某一个面试组）

由于区组数 45 达到不等式（4）上界的最优解，可见不等式是紧的。

从表一中不难看出，取 M =12，N=45 的情况，每一位文理科老师都恰好面试 $\frac{4\times 45}{12}=15$ 个学生，实现了老师面试学生数的均衡。

同理，当老师数 M =24 时，也可根据此法排列，将 M =24 代入式子：

$$N\le\left\lfloor\frac{M}{4}\cdot\left\lfloor\frac{M}{4}\right\rfloor\right\rfloor\left(\frac{M}{2}-1\right)$$

得到 N 的上界 $N_{\sup}$ =396，此时每一位老师恰好面试 66 个学生。而问题二中学生数量为 379，满足不等式：

$$36=\left\lfloor\frac{M}{4}\left\lfloor\frac{M}{4}\right\rfloor\right\rfloor\le N\le\left\lfloor\frac{M}{4}\cdot\left\lfloor\frac{M}{4}\right\rfloor\right\rfloor\left(\frac{M}{2}-1\right)=396$$

所以不可能做到两个不同学生的“面试组”中相同老师数均不超过 1，但理论上可以满足相同老师数均不超过 2。

按照上述编排方法，对于 M=24 N=396 的情况，具体分配方案如下（程序见附录四），问题三中要求的 M=24 N=379 的情况只要从下列数据中随机删除 396－379＝17 组四元组即可：

（下列数据前两个数字对为理科老师对，后面的 6 对字母对为可以与其搭配的文科老师对）

(1,2) (a,b) (c,d) (e,f) (g,h) (i,j) (k,l)
(1,3) (a,c) (b,d) (e,g) (f,h) (i,k) (j,l)
(1,4) (a,d) (b,c) (e,h) (f,g) (i,l) (j,k)
(1,5) (a,e) (b,f) (c,i) (d,j) (g,k) (h,l)
(1,6) (a,f) (b,e) (c,j) (d,i) (g,l) (h,k)
(1,7) (a,g) (b,h) (c,k) (d,l) (e,i) (f,j)
(1,8) (a,h) (b,g) (c,l) (d,k) (e,j) (f,i)
(1,9) (a,i) (b,j) (c,g) (d,h) (e,k) (f,l)
(1,10) (a,j) (b,i) (c,h) (d,g) (e,l) (f,k)
(1,11) (a,k) (b,l) (c,e) (d,f) (g,i) (h,j)
(1,12) (a,l) (b,k) (c,f) (d,e) (g,j) (h,i)

(2,3) (a,d) (b,c) (e,h) (f,g) (i,l) (j,k)
(2,4) (a,c) (b,d) (e,g) (f,h) (i,k) (j,l)
(2,5) (a,f) (b,e) (c,j) (d,i) (g,l) (h,k)
(2,6) (a,e) (b,f) (c,i) (d,j) (g,k) (h,l)
(2,7) (a,h) (b,g) (c,l) (d,k) (e,j) (f,i)
(2,8) (a,g) (b,h) (c,k) (d,l) (e,i) (f,j)
(2,9) (a,j) (b,i) (c,h) (d,g) (e,l) (f,k)
(2,10) (a,i) (b,j) (c,g) (d,h) (e,k) (f,l)
(2,11) (a,l) (b,k) (c,f) (d,e) (g,j) (h,i)
(2,12) (a,k) (b,l) (c,e) (d,f) (g,i) (h,j)

(3,4) (a,b) (c,d) (e,f) (g,h) (i,j) (k,l)
(3,5) (a,k) (b,l) (c,e) (d,f) (g,i) (h,j)
(3,6) (a,l) (b,k) (c,f) (d,e) (g,j) (h,i)
(3,7) (a,i) (b,j) (c,g) (d,h) (e,k) (f,l)
(3,8) (a,j) (b,i) (c,h) (d,g) (e,l) (f,k)
(3,9) (a,e) (b,f) (c,i) (d,j) (g,k) (h,l)
(3,10) (a,f) (b,e) (c,j) (d,i) (g,l) (h,k)
(3,11) (a,g) (b,h) (c,k) (d,l) (e,i) (f,j)
(3,12) (a,h) (b,g) (c,l) (d,k) (e,j) (f,i)

(4,5) (a,l) (b,k) (c,f) (d,e) (g,j) (h,i)
(4,6) (a,k) (b,l) (c,e) (d,f) (g,i) (h,j)
(4,7) (a,j) (b,i) (c,h) (d,g) (e,l) (f,k)
(4,8) (a,i) (b,j) (c,g) (d,h) (e,k) (f,l)
(4,9) (a,f) (b,e) (c,j) (d,i) (g,l) (h,k)

(4,10) (a,e) (b,f) (c,i) (d,j) (g,k) (h,l)
(4,11) (a,h) (b,g) (c,l) (d,k) (e,j) (f,i)
(4,12) (a,g) (b,h) (c,k) (d,l) (e,i) (f,j)

(5,6) (a,b) (c,d) (e,f) (g,h) (i,j) (k,l)
(5,7) (a,c) (b,d) (e,g) (f,h) (i,k) (j,l)
(5,8) (a,d) (b,c) (e,h) (f,g) (i,l) (j,k)
(5,9) (a,g) (b,h) (c,k) (d,l) (e,i) (f,j)
(5,10) (a,h) (b,g) (c,l) (d,k) (e,j) (f,i)
(5,11) (a,i) (b,j) (c,g) (d,h) (e,k) (f,l)
(5,12) (a,j) (b,i) (c,h) (d,g) (e,l) (f,k)

(6,7) (a,d) (b,c) (e,h) (f,g) (i,l) (j,k)
(6,8) (a,c) (b,d) (e,g) (f,h) (i,k) (j,l)
(6,9) (a,h) (b,g) (c,l) (d,k) (e,j) (f,i)
(6,10) (a,g) (b,h) (c,k) (d,l) (e,i) (f,j)
(6,11) (a,j) (b,i) (c,h) (d,g) (e,l) (f,k)
(6,12) (a,i) (b,j) (c,g) (d,h) (e,k) (f,l)

(7,8) (a,b) (c,d) (e,f) (g,h) (i,j) (k,l)
(7,9) (a,k) (b,l) (c,e) (d,f) (g,i) (h,j)
(7,10) (a,l) (b,k) (c,f) (d,e) (g,j) (h,i)
(7,11) (a,e) (b,f) (c,i) (d,j) (g,k) (h,l)
(7,12) (a,f) (b,e) (c,j) (d,i) (g,l) (h,k)

(8,9) (a,l) (b,k) (c,f) (d,e) (g,j) (h,i)
(8,10) (a,k) (b,l) (c,e) (d,f) (g,i) (h,j)
(8,11) (a,f) (b,e) (c,j) (d,i) (g,l) (h,k)
(8,12) (a,e) (b,f) (c,i) (d,j) (g,k) (h,l)

(9,10) (a,b) (c,d) (e,f) (g,h) (i,j) (k,l)
(9,11) (a,c) (b,d) (e,g) (f,h) (i,k) (j,l)
(9,12) (a,d) (b,c) (e,h) (f,g) (i,l) (j,k)

(10,11) (a,d) (b,c) (e,h) (f,g) (i,l) (j,k)
(10,12) (a,c) (b,d) (e,g) (f,h) (i,k) (j,l)

(11,12) (a,b) (c,d) (e,f) (g,h) (i,j) (k,l)

由于区组数 396 达到不等式（4）上界的最优解，可见不等式是紧的。从表一中不难看出，取 M =24，N=396 的情况，每一位文理科老师都恰好面试 $\frac{4\times 396}{24}=66$ 个学生，实现了老师面试学生数的均衡。

问题四：模型进一步讨论

为保证面试更趋于公平性，在保证面试不同考生的面试组成员不能完全相同情况下，学生数 N 与面试老师数 M 之间在满足以下关系：

① $N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3} \right\rfloor \right\rfloor$

在这种情况下，可确保任两位学生的面试组都没有两位面试老师相同的情形，即任两位学生的面试组至多只有一位相同的面试老师。

② $N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\left\lfloor \frac{M-2}{2} \right\rfloor \right\rfloor \right\rfloor$

在这种情况下，可确保任两位学生的面试组都没有三位面试老师相同的情形，即任两位学生的面试组至多只有两位相同的面试老师。

③ $\left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3} \right\rfloor \right\rfloor \le N \le \left\lfloor \frac{M}{4}\left\lfloor \frac{M-1}{3}\left\lfloor \frac{M-2}{2} \right\rfloor \right\rfloor \right\rfloor$

在这种情况下，无法满足两位学生的面试组都没有两位面试老师相同的情

形，但可以保证两位学生的面试组都没有三位面试老师相同的情形。

为了保证面试的公平性，除了以上提出的几点要求外，建议考虑：

① “面试组”男女教师的组合；

不同的“面试组”，招收学生的喜好不同，老师一般会在头脑中形成一个“理想人选”的模式，并以此评价学生。

② 面试之前可以增加一项调查表，调查学生的文理偏好及特长，可适当的调整“面试组”的文理科老师的比例

③ 决定录取与否的各因素中，考生哪个条件占多大权重，应进一步规范化和制度化。

参考文献：

[1]卢开澄，卢华明，组合数学，北京：清华大学出版社，2002.7

[2]吉庆兵，一类部分均衡不完全区组设计的构造，重庆师范学院学报，2001,9.NO.3

[3]M．H.Alsuwaiyel,算法设计技巧与分析,北京：电子工业出版社，2004.8

[4]耿素云，屈婉玲，离散数学，北京：高等教育出版社，1998.6

[5]周明 孙树栋，遗传算法原理及应用，北京：国防工业出版社 1999.6

[6]Michalewicz,Z.，演化程序——遗传算法和数据编码的结合，北京：科学出版社 ，2000.

附录

附录一

```
Program GA2;
{$APPTYPE CONSOLE}
uses
  SysUtils;
const
  N = 379;
  M = 24;
  OddFlag = true;//是否分文理
type
  StudentChoice = record
    choice: array[1..4] of SmallInt;
  end;
  Students = record
    Stu: array[1..N] of StudentChoice;
    Tea: array[1..M, 1..N] of Byte;
    ATAMatr: array[1..n, 1..n] of Word;
    Y1, Y3, Y4: Integer;
  end;
var
  pop: array[1..2] of Students;
  temp_pop: array[1..2] of Students;
  Pop_children: array[1..2] of Students;
  answer: array[1..m, 1..m] of SmallInt;
  DNAcount, GenCount, Goal: Integer;

procedure MakeAdjMat(AB: BYTE); //建立临接矩阵
var i, j: Integer;
begin
  for i := 1 to m do
    for j := 1 to n do
      temp_pop[AB].Tea[i, j] := 0;
  for i := 1 to n do
    for j := 1 to 4 do
      temp_pop[AB].Tea[temp_pop[AB].Stu[i].choice[j], i] := 1;
end;
```

```
procedure ATA(AB: Byte); //建立 AT*A 矩阵
var i, j, k: Integer;
begin
  for i := 1 to n do
    for j := 1 to n do temp_pop[AB].ATAMatr[i, j] := 0;
  for i := 1 to n do
    for j := 1 to n do
    begin
      for k := 1 to m do
        temp_pop[AB].ATAMatr[i, j] := temp_pop[AB].ATAMatr[i, j] +
temp_pop[AB].Tea[k, i] * temp_pop[AB].Tea[k, j];
    end;
end;

function Fit_Y1(AB: Byte): Integer; //方差
var i, j, aver, t, count: Integer;
  teacher: array[1..M] of Integer;
begin
  count := 0;
  aver := Round(4 * N / M);
  for i := 1 to m do teacher[i] := 0;
  for i := 1 to m do
    for j := 1 to n do
      if temp_pop[AB].Tea[i, j] = 1 then Inc(teacher[i]);
  for i := 1 to m do
  begin
    t := ABS(teacher[i] - aver);
    count := count + t;
  end;
  Result := count;
end;

function Fit_Y3(AB: Byte): Integer;
var i, j, count: Integer;
begin
  count := 0;
  for i := 1 to n do
    for j := i + 1 to n do
      count := count + Abs(temp_pop[AB].ATAMatr[i, j] - 1);
```

```
  Result := count;
end;

function Fit_Y4(AB: Byte): Integer; //两个老师的学生集合中出现相同学生的人数
begin
  Result := pop[AB].Y4;
end;

procedure Init; //总群初始化
var i, j, k: Integer;
  F: TextFile;
begin
  AssignFile(F, 'in.txt');
  Reset(F);
  DNAcount := 0;
  GenCount := 0;
  read(F, Goal);
  for i := 1 to 2 do
  begin
    for j := 1 to N do
      for k := 1 to 4 do
      begin
        read(F, temp_pop[i].Stu[j].choice[k]);
      end;
    MakeAdjMat(i);
    ATA(i);
    temp_pop[i].Y1 := Fit_Y1(i);
    temp_pop[i].Y3 := Fit_Y3(i);
    temp_pop[i].Y4 := Fit_Y4(i);
    pop[i] := temp_pop[i];
  end;
  CloseFile(F);
end;

function Compare(a, b: StudentChoice): Boolean;
var i, j, count: Integer;
  flag: Boolean;
begin
  count := 0; flag := true;
```

```
  for i := 1 to 4 do
    for j := 1 to 4 do
      if a.choice[i] = b.choice[j] then Inc(count);
  if count = 4 then flag := False;
  Result := flag;
end;

function CheckCross(p: Integer): Boolean; //判定交叉运算合法性
var a, b: StudentChoice;
  i: Integer;
  flag: Boolean;
begin
  flag := True;
  a := temp_pop[1].stu[p]; b := temp_pop[2].stu[p];
  for i := 1 to n do
    if i <> p then
    begin
      if not Compare(a, temp_pop[2].stu[i]) then flag := False;
      if not Compare(b, temp_pop[1].stu[i]) then flag := False;
      if flag = False then Break;
    end;
  Result := flag;
end;

function CheckMutant(AB: Byte; p: Integer): Boolean; //判定变异运算合法性
var i: Integer; flag: Boolean;
begin
  flag := true;
  for i := 1 to N do
    if i <> p then
    begin
      flag := Compare(temp_pop[AB].stu[i], temp_pop[AB].stu[p]);
      if not flag then Break;
    end;
  Result := flag;
end;

function CheckRandom(num: Integer): Boolean; //随机触发
var t: Integer; flag: Boolean;
```

```
begin
  flag := true;
  Randomize;
  t := Random(100) + 1;
  if t > num then flag := False;
  Result := flag;
end;

function Probability(ProType: Byte): Boolean; //概率调整
const
  CrossAver = 40; Mutant = 10;
var pro: Integer; flag: Boolean;
begin
  flag := True;
  if GenCount in [1..20] then
  begin
    if ProType = 0 then
      flag := CheckRandom(60)
    else flag := CheckRandom(40);
  end;
  if GenCount in [21..40] then
  begin
    if ProType = 0 then
      flag := CheckRandom(40)
    else flag := CheckRandom(30);
  end;
  if GenCount > 40 then
  begin
    if ProType = 0 then
      flag := CheckRandom(30)
    else flag := CheckRandom(10);
  end;
  Result := flag;
end;

procedure Cross; //交叉运算
var i: Integer;
  t: StudentChoice;
begin
```

```
  for i := 1 to N do
  begin
    if Probability(0) then
      if CheckCross(i) then
      begin
        t := temp_pop[1].stu[i];
        temp_pop[1].Stu[i] := temp_pop[2].Stu[i];
        temp_pop[2].Stu[i] := t;
      end;
  end;
end;

procedure Mutant; //变异运算
var i, j, position, value, t: Integer;
begin
  for j := 1 to 2 do
    for i := 1 to N do
      if Probability(1) then
      begin
        position := Random(4) + 1;
        value := 0;
        while (value = 0) or
          (value = temp_pop[j].Stu[i].choice[1]) or
          (value = temp_pop[j].Stu[i].choice[2]) or
          (value = temp_pop[j].Stu[i].choice[3]) or
          (value = temp_pop[j].Stu[i].choice[4]) do
        begin
          value := Random(m) + 1;
          if OddFlag then
           if  ((temp_pop[j].Stu[i].choice[position]  mod  2=0)and(  value  mod
2<>0))or
              ((temp_pop[j].Stu[i].choice[position] mod 2<>0)and( value mod 2=0))
            then Dec(value);
        end;
        t := temp_pop[j].Stu[i].choice[position]; //记录原位置数据
        temp_pop[j].Stu[i].choice[position] := value;
        if not CheckMutant(j, i) then
          temp_pop[j].Stu[i].choice[position] := t; //还原
      end;
end;
```

```
function Fit(AB: Byte): Boolean; //适应度判断
var value: Real;
  Y1, Y3, Y4: Integer;
begin
  MakeAdjMat(AB);
  ATA(AB);
  Y1 := Fit_Y1(AB);
  Y3 := Fit_Y3(AB);
  Y4 := Fit_Y4(AB);

  value := 0.5 * (Y1 / pop[1].Y1) + 0.5 * (Y3 / pop[1].Y3);
  if value < 1 then value := 0.5 * (Y1 / pop[2].Y1) + 0.5 * (Y3 / pop[2].Y3);

  if value >= 1 then Result := False
  else
  begin
    temp_pop[AB].Y1 := Y1;
    temp_pop[AB].Y3 := Y3;
    temp_pop[AB].Y4 := Y4;
    Result := True;
  end;
end;

procedure MakeChild(AB, p: Integer);
begin
  Pop_children[p] := temp_pop[AB];
end;

procedure ChooseChild;
var i: integer;
begin
  for i := 1 to 2 do
    pop[i] := Pop_children[i];
end;

procedure PrintPop;
var i, j, k, AB: Integer;
  str: string;
```

```
  f: TextFile;
begin
  for AB := 1 to 2 do
  begin
    //打印学生选择
    assign(f, 'ANS/STU/' + inttostr(GenCount) + '_' + inttostr(AB) + '.txt');
    rewrite(f);
    for j := 1 to N do
    begin
      str := '';
      for k := 1 to 4 do str := str + (IntToStr(pop[AB].Stu[j].choice[k]) + ' ');
      writeln(f, str);
    end;
    close(f);

    //打印零接矩阵
    assign(f, 'ANS/ADJ/' + inttostr(GenCount) + '_' + inttostr(AB) + '.txt');
    rewrite(f);
    for i := 1 to m do
    begin
      str := '';
      for j := 1 to n do str := str + (IntToStr(pop[AB].Tea[i, j]) + ' ');
      Writeln(f, str);
    end;
    close(f);

    //打印 A*AT
    assign(f, 'ANS/AAT/' + inttostr(GenCount) + '_' + inttostr(AB) + '.txt');
    rewrite(f);
    for i := 1 to m do
      for j := 1 to m do answer[i, j] := 0;
    for i := 1 to m do
      for j := 1 to m do
      begin
        for k := 1 to n do
          answer[i, j] := answer[i, j] + pop[AB].Tea[i, k] * pop[AB].Tea[j, k];
      end;
    for i := 1 to m do
    begin
```

```
      str := '';
      for j := 1 to m do str := str + inttostr(answer[i, j]) + ' ';
      Writeln(f, str + ';');
    end;
    close(f);
    //打印 AT*A
    assign(f, 'ANS/ATA/' + inttostr(GenCount) + '_' + inttostr(AB) + '.txt');
    rewrite(f);
    for i := 1 to n do
    begin
      str := '';
      for j := 1 to n do str := str + inttostr(pop[AB].ATAMatr[i, j]) + ' ';
      Writeln(f, str + ';');
    end;
    Close(f);
  end;
end;

procedure Go(step: Integer); //遗传主过程
var i, p, k: Integer;
begin
  for k := 1 to step do
  begin
    GenCount := k;
    DNAcount := 0;
    p := 0; //两个中有几个成功
    while p < 2 do
    begin
      Inc(DNAcount);
      Cross;
      Mutant;
      for i := 1 to 2 do
        if Fit(i) and (p < 2) then
        begin
          Inc(p);
          MakeChild(i, p);
        end;
    end;
    ChooseChild;
```

```
    Writeln(IntToStr(GenCount) + '(' + inttostr(DNAcount) + ')' + '[' +
inttostr(pop[1].Y1) + ',' + inttostr(pop[1].Y3) + ']');
    if (k mod 1 = 0) then PrintPop;
  end;
end;


begin
  { TODO -oUser -cConsole Main : Insert code here }
  Randomize;
  Init;
  Go(Goal);
  Writeln('DONE.');
  Readln;
end.
```

附录二

```
g1=[
352 202 147 38 41 36 39 38 37 41 37 39 35 30 31 29 31 28 30 30 29 28 28 32 ;
202 219 25 20 20 22 20 21 21 21 19 20 21 19 22 21 19 23 19 19 20 20 20 23 ;
147 25 163 18 22 20 21 21 21 20 19 21 15 8 11 12 13 10 10 11 11 11 11 11 ;
38 20 18 42 2 3 2 3 2 3 4 4 2 3 2 2 2 2 1 4 2 2 3 2 ;
41 20 22 2 45 2 2 5 3 3 2 2 2 2 3 2 5 2 3 2 2 3 2 3 ;
36 22 20 3 2 42 3 3 2 2 2 2 2 2 2 1 3 2 2 2 2 3 3 5 ;
39 20 21 2 2 3 42 2 2 2 3 2 2 3 2 2 2 2 2 2 2 4 3 2 ;
38 21 21 3 5 3 2 43 2 2 3 3 2 2 3 2 2 2 2 2 2 2 2 3 ;
37 21 21 2 3 2 2 2 42 2 2 3 2 2 1 4 2 4 2 3 2 2 3 2 ;
41 21 20 3 3 2 2 2 2 42 1 2 2 3 2 3 3 1 3 2 2 2 2 2 ;
37 19 19 4 2 2 3 3 2 1 42 3 3 3 2 3 5 2 3 2 2 2 2 2 ;
39 20 21 4 2 2 2 3 3 2 3 43 2 2 2 2 4 2 2 3 2 2 2 3 ;
35 21 15 2 2 2 2 2 2 2 2 3 2 36 2 2 3 2 2 2 1 1 1 1 1 ;
30 19 8 3 2 2 3 2 2 3 3 2 2 32 1 1 1 2 2 1 1 1 0 5 ;
31 22 11 2 3 2 2 3 1 2 2 2 2 1 33 2 1 2 1 1 2 1 2 1 ;
29 21 12 2 2 1 2 2 4 3 3 2 3 1 2 33 1 1 2 1 2 1 1 1 ;
31 19 13 2 5 3 2 2 2 3 5 4 2 1 1 1 36 2 2 1 1 2 3 1 ;
28 23 10 2 2 2 2 2 4 1 2 2 2 2 2 1 2 33 1 2 1 1 2 3 ;
30 19 10 1 3 2 2 2 2 3 3 2 2 2 1 2 2 1 32 1 1 1 1 3 ;
30 19 11 4 2 2 2 2 3 2 2 3 1 1 1 1 1 2 1 32 1 1 2 2 ;
29 20 11 2 2 2 2 2 2 2 2 2 1 1 2 2 1 1 1 1 31 3 1 1 ;
28 20 11 2 3 3 4 2 2 2 2 2 1 1 1 1 2 1 1 1 3 32 1 2 ;
28 20 11 3 2 3 3 2 3 2 2 2 1 0 2 1 3 2 1 2 1 1 32 1 ;
32 23 11 2 3 5 2 3 2 2 2 3 1 5 1 1 1 3 3 2 1 2 1 37 ;
];

g5=[
273 121 92 33 32 24 34 32 35 35 31 31 26 24 33 29 28 27 31 20 21 23 30 27 ;
121 175 25 14 15 18 13 16 14 19 18 24 15 18 21 23 18 22 21 16 16 16 19 23 ;
92 25 141 19 19 18 13 20 17 15 13 12 19 12 14 16 15 16 11 11 10 12 12 12 ;
33 14 19 46 2 3 2 5 4 3 3 6 2 4 5 6 5 6 2 4 3 1 1 5 ;
32 15 19 2 49 3 5 6 4 4 3 4 2 3 5 5 4 5 4 6 3 5 4 4 ;
24 18 18 3 3 46 3 3 3 3 5 3 5 3 3 2 6 4 3 5 1 5 4 11 ;
34 13 13 2 5 3 41 5 2 3 3 3 4 4 4 4 1 5 3 3 3 0 2 4 ;
```

```
32 16 20 5 6 3 5 49 7 4 3 4 4 1 4 2 3 1 4 5 2 3 5 8 ;
35 14 17 4 4 3 2 7 50 5 4 3 4 5 2 7 2 2 5 4 2 5 8 6 ;
35 19 15 3 4 3 3 4 5 51 5 5 3 4 3 8 2 7 6 5 3 2 4 5 ;
31 18 13 3 3 5 3 3 4 5 45 2 4 3 5 4 3 3 4 6 3 1 5 4 ;
31 24 12 6 4 3 3 4 3 5 2 46 2 1 3 4 6 3 5 5 2 3 3 4 ;
26 15 19 2 2 5 4 4 4 3 4 2 39 2 4 1 5 3 1 2 2 2 2 3 ;
24 18 12 4 3 3 4 1 5 4 3 1 2 40 4 2 5 6 4 2 3 3 2 5 ;
33 21 14 5 5 3 4 4 2 3 5 3 4 4 47 5 4 3 4 4 2 4 3 2 ;
29 23 16 6 5 2 4 2 7 8 4 4 1 2 5 49 3 5 4 5 2 3 4 3 ;
28 18 15 5 4 6 1 3 2 2 3 6 5 5 4 3 42 2 2 2 1 2 1 6 ;
27 22 16 6 5 4 5 1 2 7 3 3 3 6 3 5 2 45 2 2 1 3 2 5 ;
31 21 11 2 4 3 3 4 5 6 4 5 1 4 4 4 2 2 43 2 4 2 3 2 ;
20 16 11 4 6 5 3 5 4 5 6 5 2 2 4 5 2 2 2 39 3 1 2 2 ;
21 16 10 3 3 1 3 2 2 3 3 2 2 3 2 2 1 1 4 3 33 4 4 4 ;
23 16 12 1 5 5 0 3 5 2 1 3 2 3 4 3 2 3 2 1 4 35 0 5 ;
30 19 12 1 4 4 2 5 8 4 5 3 2 2 3 4 1 2 3 2 4 0 41 3 ;
27 23 12 5 4 11 4 8 6 5 4 4 3 5 2 3 6 5 2 2 4 5 3 51 ;
];

g10=[
196 64 53 23 21 16 25 25 25 23 27 32 24 28 18 25 20 23 20 13 23 16 21 23 ;
64 131 21 15 10 18 13 10 17 13 23 19 14 13 12 16 14 18 15 11 14 15 19 9 ;
53 21 112 17 8 10 12 15 22 10 15 16 16 12 11 11 10 11 10 12 14 14 3 13 ;
23 15 17 59 7 7 3 7 10 2 8 5 5 6 6 11 7 11 4 5 5 4 4 5 ;
21 10 8 7 45 7 7 4 6 3 5 4 4 5 1 7 7 5 3 4 7 3 4 3 ;
16 18 10 7 7 57 7 9 9 6 6 6 8 5 6 5 4 8 7 6 2 7 4 8 ;
25 13 12 3 7 7 50 9 7 5 5 7 8 4 2 7 4 6 4 0 3 2 4 6 ;
25 10 15 7 4 9 9 57 9 6 5 9 4 4 4 4 10 3 4 2 8 4 6 10 ;
25 17 22 10 6 9 7 9 60 4 5 5 7 8 6 4 5 6 4 2 1 6 6 6 ;
23 13 10 2 3 6 5 6 4 51 10 8 5 9 5 7 2 6 9 6 4 4 2 4 ;
27 23 15 8 5 6 5 5 5 10 59 4 6 9 1 3 6 6 5 8 6 5 4 5 ;
32 19 16 5 4 6 7 9 5 8 4 64 7 10 2 12 12 3 3 9 6 5 5 3 ;
24 14 16 5 4 8 8 4 7 5 6 7 54 4 9 3 5 5 6 2 5 4 3 8 ;
28 13 12 6 5 5 4 4 8 9 9 10 4 58 3 7 6 5 6 4 11 5 5 5 ;
18 12 11 6 1 6 2 4 6 5 1 2 9 3 40 3 3 3 3 4 1 7 5 5 ;
25 16 11 11 7 5 7 4 4 7 3 12 3 7 3 52 3 3 6 5 7 3 3 1 ;
20 14 10 7 7 4 4 10 5 2 6 12 5 6 3 3 50 5 4 6 3 4 5 5 ;
23 18 11 11 5 8 6 3 6 6 6 3 5 5 3 3 5 52 3 5 5 4 4 8 ;
20 15 10 4 3 7 4 4 4 9 5 3 6 6 3 6 4 3 46 2 7 4 3 6 ;
```

```
13 11 12 5 4 6 0 2 2 6 8 9 2 4 4 5 6 5 2 39 3 4 1 3 ;
23 14 14 5 7 2 3 8 1 4 6 6 5 11 1 7 3 5 7 3 49 4 5 3 ;
16 15 14 4 3 7 2 4 6 4 5 5 4 5 7 3 4 4 4 4 4 44 2 6 ;
21 19 3 4 4 4 4 6 6 2 4 5 3 5 5 3 5 4 3 1 5 2 41 5 ;
23 9 13 5 3 8 6 10 6 4 5 3 8 5 5 1 5 8 6 3 3 6 5 50 ;
];

g15=[
137 36 32 17 20 12 16 19 14 14 20 16 16 16 18 15 14 11 17 13 23 19 14 19 ;
36 110 19 14 10 13 13 13 19 16 19 10 8 11 15 19 11 12 12 12 8 17 8 15 ;
32 19 101 18 14 14 11 17 13 12 8 8 15 10 14 8 8 11 10 9 14 10 10 18 ;
17 14 18 62 9 9 7 7 5 4 7 4 6 6 5 8 8 5 7 6 7 13 7 7 ;
20 10 14 9 62 7 9 10 11 5 6 5 7 7 7 6 2 5 4 5 10 13 7 7 ;
12 13 14 9 7 56 9 11 6 2 9 4 10 2 8 6 5 6 6 8 6 5 4 6 ;
16 13 11 7 9 9 58 6 6 6 13 7 6 8 5 4 7 8 2 6 5 9 3 8 ;
19 13 17 7 10 11 6 63 8 6 10 6 7 6 6 6 3 6 7 5 8 8 4 10 ;
14 19 13 5 11 6 6 8 55 4 7 6 5 5 7 6 5 6 3 5 5 6 4 9 ;
14 16 12 4 5 2 6 6 4 52 7 5 6 7 6 9 3 5 7 4 5 9 7 7 ;
20 19 8 7 6 9 13 10 7 7 65 10 4 5 6 3 4 4 7 7 11 12 6 10 ;
16 10 8 4 5 4 7 6 6 5 10 52 7 3 5 11 6 12 6 5 3 8 3 6 ;
16 8 15 6 7 10 6 7 5 6 4 7 54 6 6 6 2 5 7 4 10 6 6 7 ;
16 11 10 6 7 2 8 6 5 7 5 3 6 49 7 4 8 3 7 2 10 4 3 7 ;
18 15 14 5 7 8 5 6 7 6 6 5 6 7 60 8 10 4 7 5 8 8 6 9 ;
15 19 8 8 6 6 4 6 6 9 3 11 6 4 8 53 3 7 6 6 5 5 3 5 ;
14 11 8 8 2 5 7 3 5 3 4 6 2 8 10 3 45 4 4 6 2 7 5 8 ;
11 12 11 5 5 6 8 6 6 5 4 12 5 3 4 7 4 49 6 3 7 6 7 4 ;
17 12 10 7 4 6 2 7 3 7 7 6 7 7 7 6 4 6 51 7 4 6 5 6 ;
13 12 9 6 5 8 6 5 5 4 7 5 4 2 5 6 6 3 7 46 2 4 6 8 ;
23 8 14 7 10 6 5 8 5 5 11 3 10 10 8 5 2 7 4 2 59 9 8 7 ;
19 17 10 13 13 5 9 8 6 9 12 8 6 4 8 5 7 6 6 4 9 66 7 7 ;
14 8 10 7 7 4 3 4 4 7 6 3 6 3 6 3 5 7 5 6 8 7 46 5 ;
19 15 18 7 7 6 8 10 9 7 10 6 7 7 9 5 8 4 6 8 7 7 5 65 ;
];

g20=[
106 16 15 13 16 14 16 15 17 11 14 8 9 11 14 19 12 12 14 18 14 17 10 13 ;
16 86 18 7 9 12 11 11 13 9 9 13 4 11 11 12 15 10 13 9 9 11 13 12 ;
15 18 77 9 12 15 10 9 10 9 6 7 5 5 11 7 13 10 7 9 12 9 11 12 ;
13 7 9 60 8 4 6 3 9 7 7 10 9 6 12 6 8 6 8 9 6 9 8 10 ;
```

16 9 12 8 66 9 13 9 8 7 6 9 7 8 9 6 12 7 7 7 8 8 9 4 ;
14 12 15 4 9 63 9 9 11 4 9 10 7 4 5 7 8 6 6 6 9 7 7 11 ;
16 11 10 6 13 9 68 5 8 8 13 6 9 10 6 8 9 8 7 9 8 10 7 8 ;
15 11 9 3 9 9 5 52 4 3 7 4 3 5 5 5 6 7 10 5 6 7 5 13 ;
17 13 10 9 8 11 8 4 70 11 13 6 9 11 6 15 8 8 7 6 11 8 6 5 ;
11 9 9 7 7 4 8 3 11 52 7 6 8 6 4 6 4 7 10 6 10 7 1 5 ;
14 9 6 7 6 9 13 7 13 7 58 9 5 6 2 6 8 10 7 7 4 7 8 4 ;
8 13 7 10 9 10 6 4 6 6 9 53 5 6 9 5 6 5 8 5 3 5 4 10 ;
9 4 5 9 7 7 9 3 9 8 5 5 51 4 7 7 4 5 8 6 8 7 6 11 ;
11 11 5 6 8 4 10 5 11 6 6 6 4 52 6 4 5 6 6 2 4 8 9 13 ;
14 11 11 12 9 5 6 5 6 4 2 9 7 6 57 8 8 3 9 7 9 8 4 8 ;
19 12 7 6 6 7 8 5 15 6 6 5 7 4 8 58 5 7 6 7 10 6 8 4 ;
12 15 13 8 12 8 9 6 8 4 8 6 4 5 8 5 63 8 7 15 4 12 5 7 ;
12 10 10 6 7 6 8 7 8 7 10 5 5 6 3 7 8 59 8 9 9 9 8 9 ;
14 13 7 8 7 6 7 10 7 10 7 8 8 6 9 6 7 8 61 5 8 7 5 10 ;
18 9 9 9 7 6 9 5 6 6 7 5 6 2 7 7 15 9 5 59 9 10 5 6 ;
14 9 12 6 8 9 8 6 11 10 4 3 8 4 9 10 4 9 8 9 60 5 7 7 ;
17 11 9 9 8 7 10 7 8 7 7 5 7 8 8 6 12 9 7 10 5 64 9 6 ;
10 13 11 8 9 7 7 5 6 1 8 4 6 9 4 8 5 8 5 5 7 9 55 10 ;
13 12 12 10 4 11 8 13 5 5 4 10 11 13 8 4 7 9 10 6 7 6 10 66 ;
];

g25=[
63 11 3 14 2 8 5 10 7 10 8 11 5 15 1 10 9 13 7 6 9 14 7 4 ;
11 65 14 6 11 8 14 10 14 6 10 6 8 6 12 3 11 5 8 8 9 4 8 3 ;
3 14 64 10 8 16 7 13 5 9 3 5 8 10 3 11 6 10 8 8 5 12 8 10 ;
14 6 10 63 9 5 9 6 9 7 17 4 13 7 6 8 12 6 10 6 10 5 7 3 ;
2 11 8 9 61 11 4 9 5 13 6 7 5 9 7 12 5 10 7 15 5 11 7 5 ;
8 8 16 5 11 64 7 6 5 5 11 7 6 6 9 5 14 3 14 8 13 8 14 3 ;
5 14 7 9 4 7 60 9 8 11 6 9 4 15 2 9 6 12 5 7 8 10 5 8 ;
10 10 13 6 9 6 9 66 8 9 9 6 8 4 15 4 18 3 10 5 14 7 9 6 ;
7 14 5 9 5 5 8 8 63 11 8 14 2 11 7 10 6 12 3 13 6 10 6 9 ;
10 6 9 7 13 5 11 9 11 69 14 8 8 6 11 9 7 5 11 3 15 4 18 7 ;
8 10 3 17 6 11 6 9 8 14 64 10 9 11 4 13 4 8 7 9 5 8 4 8 ;
11 6 5 4 7 7 9 6 14 8 10 61 7 4 11 6 11 5 12 6 11 6 14 3 ;
5 8 8 13 5 6 4 8 2 8 9 7 55 14 10 13 5 8 3 9 2 12 2 4 ;
15 6 10 7 9 6 15 4 11 6 11 4 14 71 11 7 12 8 12 6 14 14 8 3 ;
1 12 3 6 7 9 2 15 7 11 4 11 10 11 64 11 7 14 5 13 11 7 7 8 ;
10 3 11 8 12 5 9 4 10 9 13 6 13 7 11 66 9 9 9 6 13 4 12 5 ;

9 11 6 12 5 14 6 18 6 7 4 11 5 12 7 9 64 6 6 8 5 13 5 7 ;
13 5 10 6 10 3 12 3 12 5 8 5 8 8 14 9 6 63 8 9 12 7 13 3 ;
7 8 8 10 7 14 5 10 3 11 7 12 3 12 5 9 6 8 65 16 8 12 6 8 ;
6 8 8 6 15 8 7 5 13 3 9 6 9 6 13 6 8 9 16 61 10 1 8 3 ;
9 9 5 10 5 13 8 14 6 15 5 11 2 14 11 13 5 12 8 10 71 13 7 8 ;
14 4 12 5 11 8 10 7 10 4 8 6 12 14 7 4 13 7 12 1 13 65 8 5 ;
7 8 8 7 7 14 5 9 6 18 4 14 2 8 7 12 5 13 6 8 7 8 64 9 ;
4 3 10 3 5 3 8 6 9 7 8 3 4 3 8 5 7 3 8 3 8 5 9 44 ;
];
g30=[
62 6 9 9 9 5 8 5 10 6 8 11 10 4 11 12 10 8 5 12 8 6 5 9 ;
6 61 4 5 10 5 8 13 7 3 9 8 9 9 12 10 5 7 6 13 8 7 6 13 ;
9 4 67 13 12 11 8 8 8 11 9 12 5 8 7 7 12 10 7 4 11 9 7 9 ;
9 5 13 62 10 9 6 11 7 4 8 6 9 7 9 10 7 8 8 8 6 9 10 7 ;
9 10 12 10 68 5 9 7 7 10 8 9 8 8 11 9 4 13 6 8 13 12 8 8 ;
5 5 11 9 5 59 14 6 5 12 6 6 9 7 9 8 8 6 8 12 3 7 6 10 ;
8 8 8 6 9 14 58 8 8 6 6 8 7 9 8 4 10 10 5 3 6 7 4 12 ;
5 13 8 11 7 6 8 64 10 6 7 6 5 4 9 8 16 9 13 9 6 8 10 8 ;
10 7 8 7 7 5 8 10 67 11 8 11 8 11 10 7 8 10 6 9 11 11 9 9 ;
6 3 11 4 10 12 6 6 11 61 8 9 9 15 9 5 10 8 6 7 9 5 9 5 ;
8 9 9 8 8 6 6 7 8 8 66 8 14 12 7 8 12 10 6 10 3 8 12 11 ;
11 8 12 6 9 6 8 6 11 9 8 60 2 8 8 9 9 8 7 8 10 8 4 5 ;
10 9 5 9 8 9 7 5 8 9 14 2 57 11 7 7 3 7 5 7 8 4 7 10 ;
4 9 8 7 8 7 9 4 11 15 12 8 11 67 6 8 10 6 12 9 9 8 9 11 ;
11 12 7 9 11 9 8 9 10 9 7 8 7 6 67 11 9 10 7 12 10 6 5 8 ;
12 10 7 10 9 8 4 8 7 5 8 9 7 8 11 63 7 9 9 8 9 10 9 5 ;
10 5 12 7 4 8 10 16 8 10 12 9 3 10 9 7 63 8 6 7 7 5 9 7 ;
8 7 10 8 13 6 10 9 10 8 10 8 7 6 10 9 8 67 9 8 10 9 7 11 ;
5 6 7 8 6 8 5 13 6 6 6 7 5 12 7 9 6 9 60 9 6 9 13 12 ;
12 13 4 8 8 12 3 9 9 7 10 8 7 9 12 8 7 8 9 64 6 9 9 5 ;
8 8 11 6 13 3 6 6 11 9 3 10 8 9 10 9 7 10 6 6 61 6 10 8 ;
6 7 9 9 12 7 7 8 11 5 8 8 4 8 6 10 5 9 9 9 6 61 10 10 ;
5 6 7 10 8 6 4 10 9 9 12 4 7 9 5 9 9 7 13 9 10 10 63 11 ;
9 13 9 7 8 10 12 8 9 5 11 5 10 11 8 5 7 11 12 5 8 10 11 68 ;
];

附录三

19 7 14 17	1 12 7 9	7 13 5 18
13 12 11 15	18 14 10 23	18 23 19 7
1 5 22 21	7 12 10 16	8 19 5 21
21 6 22 2	3 16 9 14	8 4 2 5
1 21 12 18	14 19 23 10	22 7 24 15
5 12 7 18	9 12 3 1	9 20 13 10
17 1 23 4	23 13 20 5	14 12 18 16
19 5 15 16	22 11 14 23	24 1 9 2
24 18 22 2	5 6 15 17	20 16 14 2
22 4 5 11	24 14 19 17	7 3 24 1
24 9 14 13	2 8 16 13	17 15 1 20
23 2 20 11	12 21 3 9	11 18 3 24
21 13 3 9	22 17 7 20	14 4 11 23
20 22 23 14	16 18 4 3	6 12 20 11
22 19 23 3	23 11 24 18	4 11 3 9
22 1 13 11	7 17 16 10	17 6 18 24
5 20 22 10	9 21 7 18	4 21 19 17
19 16 24 2	18 1 11 7	20 1 18 8
24 22 2 3	3 22 5 18	14 18 15 19
16 5 3 2	11 24 15 14	9 11 15 1
11 14 13 20	4 1 15 22	22 2 11 8
21 10 12 17	4 19 5 14	12 9 19 10
18 13 10 12	6 15 24 7	22 19 12 17
3 7 14 17	18 8 3 10	5 18 20 11
11 21 4 6	23 6 8 10	24 10 1 13
17 20 11 8	17 18 11 14	20 5 18 3
6 12 5 10	1 18 21 3	22 2 18 15
4 1 16 5	23 1 20 9	14 9 24 1
2 24 10 5	13 14 21 20	19 8 24 20
12 17 21 9	16 1 15 2	3 11 10 17
8 19 20 9	16 9 23 22	6 23 22 24
4 12 20 1	21 14 10 1	9 13 11 8
2 7 17 8	23 7 11 5	10 24 7 6
9 7 24 15	7 6 16 9	2 12 20 16
8 4 12 17	14 9 13 21	2 8 9 21
24 11 5 22	6 19 20 22	5 21 24 18
8 4 13 1	4 10 12 8	1 3 7 12

6 16 20 8
7 15 20 6
14 23 13 4
23 8 22 17
19 5 1 18
20 6 24 4
11 13 16 15
13 23 17 6
16 4 22 5
21 24 23 3
19 22 12 16
16 22 21 23
23 12 21 24
17 12 6 8
18 14 2 11
14 15 20 8
6 7 16 4
24 21 14 6
2 5 11 16
4 13 5 21
6 3 10 14
5 3 21 12
9 21 13 2
24 11 16 1
10 18 17 8
20 16 22 19
15 10 9 12
9 18 4 22
21 8 20 15
6 1 19 12
11 16 14 4
18 7 8 2
22 5 20 18
3 9 22 18
19 24 8 21
22 3 4 6
7 4 1 24
3 19 24 12
19 11 23 22

18 22 8 19
13 18 7 24
9 14 10 19
22 14 15 9
12 16 15 23
21 16 23 8
16 1 15 6
21 13 5 24
23 7 4 9
8 9 17 15
14 19 7 24
19 18 4 24
14 11 12 20
15 13 6 10
12 7 17 24
15 22 4 3
19 13 15 4
10 14 16 22
15 13 2 14
8 23 15 4
2 20 15 23
9 22 15 1
12 1 21 7
17 14 5 2
19 18 11 13
1 9 20 10
12 20 2 15
15 18 16 21
10 21 18 23
16 24 1 15
2 15 19 8
18 8 17 9
6 7 13 17
21 10 20 15
2 24 13 5
7 11 2 9
11 24 23 2
4 18 3 24
22 11 7 12

5 10 7 3
8 4 19 23
19 9 6 14
10 14 3 13
19 8 24 4
5 14 6 10
5 20 19 21
20 6 2 14
18 15 24 2
22 12 14 10
15 3 12 2
3 4 12 5
9 2 4 16
23 19 13 24
3 23 19 11
10 9 14 20
8 22 17 16
5 16 21 3
4 11 14 12
13 16 11 3
7 14 5 8
9 5 8 4
11 10 6 18
10 11 8 17
24 23 21 14
3 15 4 7
11 9 5 24
15 5 3 8
20 13 16 14
23 11 5 21
14 16 24 1
6 7 15 3
13 11 7 1
2 13 5 1
9 24 5 22
23 3 6 10
17 14 10 21
3 18 8 13
21 14 7 5

20 16 2 19
12 22 24 9
9 11 20 19
15 14 10 19
15 17 4 13
18 13 16 1
8 11 15 17
5 15 17 7
24 4 20 23
23 1 17 8
11 24 13 1
13 4 11 14
2 7 20 6
13 16 11 23
2 23 13 10
15 5 17 23
21 2 8 16
7 6 17 2
18 11 22 16
6 24 19 2
22 9 14 12
13 9 6 4
22 12 20 1
11 20 10 6
20 2 13 15
13 4 7 6
24 13 8 22
23 4 20 19
23 1 16 19
21 3 12 7
24 6 18 7
18 23 22 19
12 1 2 15
12 16 1 5
6 7 3 14
18 19 14 12
17 6 10 11
17 3 14 22
24 23 14 5

22 9 10 21
23 11 3 17
19 10 2 7
9 17 8 3
20 17 9 11
5 9 10 12
17 20 15 16
24 15 19 16
18 17 11 16
21 17 10 23
4 1 17 3
8 11 3 6
8 19 1 17
21 15 16 4
17 11 3 2
3 23 21 19
21 18 15 24
7 14 24 11
13 1 5 6
3 20 21 4
8 9 17 14
6 9 18 16
5 18 21 16
13 1 5 23
9 21 23 15
8 7 22 4
18 9 23 20
11 17 1 20
16 9 4 18
22 20 1 3
19 23 6 20
10 18 5 15
8 9 5 22
21 16 19 22
11 15 10 9
11 1 4 10
22 21 12 2
18 15 4 6
19 24 7 8

24 6 17 20
21 14 7 1
12 6 3 20
21 11 15 2
8 17 3 24
24 6 22 4
3 16 4 17
9 15 7 21
21 18 17 1
1 17 19 3
7 18 17 9
24 3 10 8
20 2 8 1
20 1 15 5
3 1 6 16
2 20 4 5
4 3 17 23
2 12 18 21
2 9 24 23
24 2 13 14
8 7 2 5
2 14 9 12
9 16 10 23
2 12 17 14
23 6 19 9
21 5 14 10
8 16 12 23
5 4 15 18
14 4 10 3
12 23 11 5
24 17 10 3
5 10 3 6
21 2 5 15
23 5 7 22
13 16 10 6
21 13 3 1
18 9 12 5
14 6 19 3
20 2 1 4

20 18 8 15
10 1 9 15
19 14 11 13
21 10 3 4
10 13 19 11
12 20 4 18
15 20 10 18
18 19 5 3
15 5 1 10
5 3 12 15
11 3 12 8
18 2 8 23
2 24 22 20
13 7 22 10
16 12 8 19

8 4 19 2
22 3 9 5
16 14 22 5
9 8 23 24
9 20 21 13
17 1 16 12
24 17 11 12
21 17 16 15
1 6 20 16
16 8 4 23
9 1 18 17
14 2 8 7
7 18 6 13
8 19 15 6
6 8 7 22

23 17 10 14
6 13 4 22
11 24 13 2
15 6 12 3

附录四

```
Program Project1;
{$APPTYPE CONSOLE}
uses
  SysUtils;
const
  Max = 66;
type
  Teacher = array[1..2] of Integer;
  GirdUnit = record
    TeaA, TeaB: Teacher;
    Adj: Boolean;
  end;
var
  line, N, half, num: Integer;
  Tea_Arr: array[1..Max] of Teacher;
  Gird: array[1..max, 1..Max] of Byte;

procedure init;
var i, j, count: Integer;
begin
  half := N div 2;
  line := (half * (half - 1)) div 2;
  num := N div 4 - 1;
  for i := 1 to line do
    for j := 1 to line do
      Gird[i, j] := 0;
  count := 0;
  for i := 1 to half do
    for j := i + 1 to half do
    begin
      Inc(count);
      Tea_Arr[count][1] := i;
      Tea_Arr[count][2] := j;
    end;
end;

procedure print;
var i, j, k, a, b, goal: Integer;
begin
```

```
  for i := 1 to half - 1 do Gird[i, i] := 1;
  for i := 1 to line do
  begin
    Write('(', Tea_Arr[i][1], ',', Tea_Arr[i][2], ') ');
    a := Tea_Arr[i][1];
    b := Tea_Arr[i][2];
    for j := 1 to line do
      if (Tea_Arr[j][1] = a) and (Tea_Arr[j][2] = b) then
        for k := 1 to half - 1 do if Gird[k, j] = 1 then
          begin
            goal := k;
            Break;
          end;
    for j := 1 to line do
      if Gird[goal, j] = 1 then write('(', chr(96 + Tea_Arr[j][1]), ',', chr(96 +
Tea_Arr[j][2]), ') ');
    Writeln;
  end;
end;

procedure doit;
var i, count: Integer;

  procedure Go(p: Integer);
  var j, k: Integer; flag: Boolean;
  begin
    for j := p + 1 to line do
      if count < num then
      begin
        flag := true;
        for k := 1 to j - 1 do
          if Gird[i, k] = 1 then
            if (Tea_Arr[k][1] = Tea_Arr[j][1]) or
              (Tea_Arr[k][1] = Tea_Arr[j][2]) or
              (Tea_Arr[k][2] = Tea_Arr[j][1]) or
              (Tea_Arr[k][2] = Tea_Arr[j][2]) then flag := False;
        if (Tea_Arr[i][1] = Tea_Arr[j][1]) or
          (Tea_Arr[i][1] = Tea_Arr[j][2]) or
          (Tea_Arr[i][2] = Tea_Arr[j][1]) or
          (Tea_Arr[i][2] = Tea_Arr[j][2]) then flag := False;
```

```
        if flag then
        begin
          flag := True;
          for k := 1 to i - 1 do
            if Gird[k, j] = 1 then
            begin
              flag := False;
              Break;
            end;
          if flag then
          begin
            Gird[i, j] := 1;
            Inc(count);
            if count < num then
            begin
              Go(j);
              if (count < num) then
              begin
                Dec(count);
                Gird[i, j] := 0;
              end;
            end;
          end;
        end;
      end;

begin
  i := 1;
  while i <= half - 1 do
  begin
    count := 0;
    Go(1);
    if (i = half - 1) then
    begin
      Print;
      Halt;
    end;
    Inc(i);
```

```
  end;
end;

begin
  { TODO -oUser -cConsole Main : Insert code here }
  assign(input, 'in.txt');
  reset(input);
  assign(output, 'out.txt');
  rewrite(output);
  Readln(N);
  init;
  doit;
  Readln;
  close(input);
  close(output);
end.
```